DOCUMENT RESUME

ED 095 908 IR 001 091

AUTHOR Lippey, Gerald
TITLE A Comprehensive Question Retrieval Application to Serve Classroom Teachers.
INSTITUTION International Business Machines Corp., San Jose, Calif.
PUB DATE 73
NOTE 14p.; For related documents see IR 001 090 and 092

EDRS PRICE MF-$0.75 HC-$1.50 PLUS POSTAGE
DESCRIPTORS *Computer Oriented Programs; *Computer Programs; Computers; Individualized Instruction; Individual Tests; *Item Banks; Item Sampling; *Teachers; *Test Construction; Tests; Test Selection
IDENTIFIERS *Classroom Teacher Support System; CTSS

ABSTRACT

Classroom Teacher Support System (CTSS) is a prototype system for computer-assisted construction and scoring of tests, using questions from a central data bank. The computer programs for CTSS were developed by International Business Machines Corporation for study in the Los Angeles City Unified School District in 1969 and 1970. In order to provide maximum service to teachers, CTSS coordinators act as an interface between teachers and the computer center and maintain the test item bank. The system design, files, computer runs, and backup and recovery procedures are described in detail. (PF)

A Comprehensive Question Retrieval Application to Serve Classroom Teachers

Gerald Lippey
San Jose

INTRODUCTION

CTSS (Classroom Teacher Support System) was developed to aid teachers. The concept consists of retrieving questions according to specified attributes from a centralized data bank, assembling them into tests or exercises, and scoring student answers. Since scoring mark-sense answer sheets is a well-understood and wide-spread application, the emphasis was placed on solving systems problems related to producing lists of questions which meet the teacher's needs as he perceives them. To achieve this, the system permits items to be classified along several dimensions so that they can be selected by the computer according to criteria set by the teacher requesting a test. (The word "test" is used here to designate a list of questions, regardless of how it is to be used by the teacher who receives it.)

CTSS enables many teachers to share a collection of questions; thus, they all benefit from the advantages of specialization. Such an application has the potential of providing a teacher with access to high-quality questions; freedom from the clerical chores of test construction and scoring; a new test, tailored to his needs, for each occasion; and comparative data based on previous student responses.

Exploration of this concept began in IBM's Advanced Systems Development Division in 1968. In 1969, a joint study agreement was reached between IBM's Systems Development Division and the Los Angeles City Unified School District to develop a prototype application. System functions were specified jointly: IBM developed the computer programs, and the school district prepared an initial collection of 8000 questions in U.S.history and took responsibility for all operational aspects.

Objectives of the joint study were to

1. Confirm decisions related to functional system operation, e.g., communications procedures, forms, reports, retrieval options, item revision procedures, test modification provisions.

2. Identify problems associated with development of item pools, e.g., item classification, cost of preparation, adequacy.

3. Discover how classroom teachers would use questions when they were conveniently available, e.g., testing, drill, discussion.

4. Gain quantitative information on usage, e.g., frequency of use, length of tests, requirement for data bank size.

During the first half of 1969, functional specifications were established. Programs were coded during the second half, while the first item collection was being prepared. Systems testing began in January 1970, with six teachers in one school. CTSS slowly phased into operational use as teachers at various schools have been gradually added during the last couple of years. There are now over 200 history teachers using the system in Los Angeles schools, and several other educational institutions have installed it.

TEACHER SERVICE

CTSS is intended to be entirely under the teacher's control. It may be used or not as the teacher sees fit. The system is free of any particular philosophy of testing or other use; it is the teacher's prerogative to use it in any fashion that satisfies his needs. Questions have been used for quizzes, homework assignments, final exams, drill, review, classroom discussion, and material for special student projects.

Although it can be used with essay and short-answer questions as well, CTSS was intended for objectively scorable (multiple-choice, true-false, and matching) questions. This decision was made to encourage machine-scoring in order to collect data to help identify unsatisfactory items. Within this framework, some features were

included to accommodate item format variations: Items consisting of several questions preceded by a paragraph or table are acceptable. Also, special print control provisions permit item authors to specify overprinting of text lines (e.g., for underlining words) and to control the splitting of long items between test pages.

Item collections are maintained on disk storage. Teachers submit requests for questions on optically scanned test-request forms, which are sent directly to the computer center through the district's internal mail. This input is batched, run each night, and the resulting tests placed in the school mail the next day. Scoring is handled in the same fashion.

Item Specification

During the design of CTSS, primary attention was given to item selection. So teachers can conveniently construct tests which meet their needs, the system permits questions to be classified in several ways. Although specific questions can be requested, teachers usually request questions by specifying attributes associated with them.

Questions in a broad subject matter area (an "item collection") are classified at the least into major subject matter "categories" and at most along four additional dimensions. The category classification may be based on behavioral objectives or not, depending upon the item collection designer's wishes. It may also be structured in hierarchical levels. During the retrieval process, items are selected from both those in the category specified and those in all categories subordinate to the one specified. There may be up to five hierarchical levels in the category classification defined.

Other classification dimensions can include an assigned difficulty level, behavior level (knowledge vs. application of knowledge), keywords, and several special flags. Some dimensions (e.g., keywords) permit the item classifier to assign more than one value to each item. Dimensions which can have a large number of values (e.g., category) are coded numerically, so that, with the aid of an index, they can easily be specified on an optically scanned test-request form.

Specifications for questions are entered in "request blocks" on the test-request form. Each block consists of several fields in which the teacher specifies the attributes and the quantity of a group of items desired. While items are normally selected by attribute, a request block may be used to specify the unique identification number of an individual item desired. Thus, a test may be constructed which contains a specified number of questions in each of several categories with the desired mix of other characteristics, as well as some specific items which the teacher knows from experience and wants to include.

Tests

The result of processing a test request is a list of questions identified by the teacher's employee number and a two-digit test number assigned for that teacher by the system. The test thus produced is stored by CTSS and labeled "generation 1."

The teacher may then modify the test by requesting the system to add or delete items. To accomplish this, a test-request form is filled out which references the test number and specifies the items to be added in the same way that an initial test is requested. Items to be deleted are indicated on another field in the form. A new list of questions with the same test number will be created and labeled "generation 2." This process can be repeated until the list of questions satisfies the teacher. Only the most recent generation of a test is remembered by the system. A teacher may have up to twenty such tests retained simultaneously.

The teacher may specify on the test-request form that the test be printed on a reproduction master. He may also request up to nine different versions of the test with the items appearing in a different sequence on each.

Each time a test is printed, two associated reports are produced. An Item Characteristics report provides the answer key and informs the teacher how each item has been classified for retrieval. It may also provide references to two resources which contain information related to the content of each item. The second report repeats the teacher's request and indicates the items retrieved in response to each request block.

CTSS will score student answer sheets when the teacher so desires. Since the test has been remembered by the system, it is not necessary for a teacher to submit a scoring key. Several scoring options are available for identifying students, suppressing or adding questions, and partitioning reports. The usual scoring reports are sent to teachers. Scoring procedures and reports will not be discussed.

The system was designed so that on-going, everyday service could be provided without the need for judgement by anyone other than the teacher concerned. Probably the most important consequence of this objective was the attempt to anticipate input errors of various kinds and, whenever possible, respond automatically by addressing an explanatory message directly to the teacher.

The teacher service described above is supported in several ways. Direct daily support is provided by the internal mail service and the data processing center. At the center, request forms are batched and read by an optical mark reader; processing is accomplished at night; and output to each teacher is manually matched to its request form prior to its return to the teacher.

CTSS is, however, not administered by the data processing group. Rather, the data processing center performs a service function, while operations are monitored and managed by education-oriented personnel referred to as CTSS "coordinators." This arrangement dictates that the points of contact between the data processing center and others be well defined, so that the computer center can regard all CTSS jobs as routine production. Consequently, input from, and output to, both teachers and coordinators is handled according to well-established procedures.

Service Support

Coordinators have two areas of responsibility--one related to operational teacher services and the other related to item pools. In the teacher service area, new users of the system may obtain coordinator assistance getting started and, subsequently, in understanding errors that they make. Coordinators also supply teachers with the optically scanned request forms.

Since a user identifies himself to CTSS on the test and scoring request forms by employee number only, a file of teacher names and locations, the "teacher file," is required to automatically address the title page of tests and scoring reports. A teacher must be registered on this file prior to using CTSS. One chief responsibility of the coordinators is maintenance of the teacher file.

The file in which tests are stored, called the "active list," can also be influenced by coordinators. When a new test is generated, it is automatically added to the active list; when a test is scored, it is deleted. Since many tests are never scored by CTSS, the active list would continue to grow indefinitely unless old tests were removed. Old tests are identified by assigning an "activity date" to each test when it is created. This is reset to the current date whenever a new generation is produced or a scoring request is not successfully processed due to input error. The activity date is used to purge old unscored tests from the file. A "time-out cancellation" program removes from the active list tests whose activity dates precede a cancellation date set by a coordinator. When a test is timed-out, a notice is sent to the teacher concerned, informing him that it is no longer available for modification or for scoring against the answer key retained in the system. Time-out cancellation is initiated periodically by coordinators.

As tests are produced and scored, statistics on system activity are accumulated. A "system statistics file" contains counters which cumulate data for two-dozen kinds of user activity. For example, the number of test requests rejected due to input errors, the average number of generations produced, and the average number of questions requested on tests are accumulated. The system statistics file contains this information for two durations (a long and a short time period); the data is maintained for each item collection; and it is classified according to which of 13 teacher groups the corresponding users belong. Coordinators may reset the system statistics file counters when they wish the accumulation process to begin again.

In addition to direct contact with teachers, coordinators monitor system usage by looking over summaries produced by each test production and scoring run. There is also a report generated by the time-out cancellation program, which summarizes the number of tests in the active list, tests having scrambled versions, and tests removed from the active list by the time-out routine. Longer term activity is observed by drawing activity reports from the system statistics file described above.

Item Pool Support

The second area of coordinator responsibilities involves the item collections themselves. To use an item pool, teachers must understand how it has been classified, and they must have access to at least the index which defines subject matter category numbers and perhaps to other indexes which have been constructed. Coordinators are responsible for communicating this type of information to teachers.

Item collection maintenance is another important coordinator job. Typically, large collections of questions are made available before they have been thoroughly edited and field tested (otherwise, the development investment would be too large). Thus, one begins with relatively poor-quality items and depends upon a long range revision process during usage to improve the questions. As teacher comments and scoring data become available, items are repaired. Teacher reactions to poor-quality items, while negative, have not turned out to be a serious problem. Indeed, teachers sometimes appear to experience satisfaction when they discover and report items needing correction.

The best source of item revision information appears to be the teacher. A second source has been provided in CTSS by cumulating item usage data in an "item statistics file," associated with each item collection. The item statistics file retains information on the number of times each question appeared in a test, or was deleted from a test or suppressed from scoring by a teacher. The file also contains student-supplied data obtained when questions are machine-scored, such as the number of responses to each option and a central tendency for discrimination index.

A coordinator can obtain an "item statistics report" based on information in the item statistics file. The philosophy behind this report is that by appropriate selection of item statistics thresholds he can obtain a list of those items most likely to need revision. A coordinator may set thresholds on high teacher rejection rate, low average discrimination, unusually heavy use of a distractor, and very high or low measured difficulty level. He may also require that some minimum number of tests has been drawn, or answer sheets scored, to cause an item to be eligible for these tests.

Finally, it is the coordinators' responsibility to oversee the creation and supervise the installation of new item collections. When a new item pool is to be constructed, many decisions need to be made: a character set, i.e., those characters which are to be allowed in item text, must be chosen; the kinds of items which are permissible must be determined; the dimensions of item classification must be designed; and the method of transmitting all the necessary classification information to teachers must be planned.

There appears to be significant value in having large item pools. Teachers do not usually wish to encounter the same few items over and over. A large collection is especially useful when multiple tests are requested to cover the same material. It also enables the collection designer to include several approaches to subject matter; this is essential if the collection is to be shared by users having a variety of pedagogical styles. Finally, it appears to be helpful if teachers regard an item collection as essentially infinite in size and constantly changing, and do not, therefore, have a desire to deal manually with the entire collection at once. Experience from CTSS indicates that about 30 questions per class hour should be regarded as a minimum item pool size; 50, as more desirable; and, perhaps, about 70 as the number beyond which the cost begins to exceed the value.

SYSTEM DESIGN

CTSS was designed as a prototype because it addresses a new application area. Many of the design decisions were influenced by this. Perhaps the most obvious effect on system design was the effort to include features whose utility was questionable in order to establish their value through experience.

On the other hand, an effort was made to design low cost into the system framework. Thus, on-line terminals were rejected in favor of internal mail service, and keypunching of teacher requests is avoided by using optically scanned input forms. Costly human intervention is further reduced by sending error messages directly to teachers. Also, teachers are discouraged from requesting reproduction masters or scrambled versions unnecessarily by preventing further modification of a test once either of these more expensive printouts has been produced.

Program design priorities (from high to low) were as follows: (1) ease of coding and testing, (2) ease of modification and maintenance, (3) low storage requirements, (4) low execution time. Prototype design specifications included handling several item collections with about 10,000 items in each. The system was programmed in PL/I to run under the IBM/360 Operating System in a 74K byte partition.

A highly modular programming approach was chosen. Communication between programs is accomplished through files which are either permanently established or used as temporary interfaces. Dividing functions into a series of separately executable programs simplified programming and testing. More importantly, this made it easier to modify the system, particularly when additional features were later inserted. The permanently established files will be outlined next, followed by a brief summary of the major runs available.

Files

CTSS includes two types of permanent files: those which are item collection-independent and those which are item collection-dependent. An "item collection-independent" file is required by the system only once, irrespective of how many item collections are supported. An "item collection-dependent" file is required to be present for each item collection. The main permanent files and their contents are itemized in Tables I and II.

Table I. Principal Item Collection-Independent Files

File Name	Contents
Course File	Record for each item collection: identification, location, and parameters
Teacher File	Record for each teacher: identification and address; identification of each active test, its activity date, and its location in the active list

File Name	Contents
Active List	Record for each active test: identification of items, answer key, and status information; identification of last 50 items deleted during previous modifications; location of version file record, if any
Version File	Record for each active test having scrambled versions: scrambling keys
System Statistics File	Record for each item collection: two sets of system usage information for each teacher group

Table II. Principal Item Collection-Dependent Files

File Name	Contents
Classification File	Record for each item: item attributes and location in the item file
Item File	Item text
Item Statistics File	Record for each question: item usage data

Access to Items

Item manipulation is, of course, the key element of a test construction system. Ease and efficiency of item retrieval and item revision depend upon the item file organization. In CTSS, all of the selection decisions concerning which items are to appear on a test are made by consulting the classification file. This file contains item attributes, but no item text, and is therefore very small and easily referenced compared to the item file. During the run that produces tests, the item file is referenced only when it is necessary to format the test for printing. (Similarly, the active list contains records of specific tests by recording item identifications instead of the item text itself.)

It was decided early in the design phase that, while several dimensions of item specification would be available to teachers, one important dimension would be emphasized and retrieval optimized around it. As a result, items are ordered by subject matter category number in the item collection-dependent files, making hierarchical selection in this dimension easy to implement. Instead of building and maintaining inverted files for other dimensions,

the classification file is simply scanned within the category designated for items which meet other specified criteria. To discourage teachers from too often requesting a scan of all items in a collection for those items which have some other attribute, they are prevented from initiating such a search with a single request block. By requiring that teachers enter a category number in each request block used and limiting the range of a search initiated by one request block to the highest level of the hierarchical category classification system, CTSS can require that several request blocks be used to cause a search over the entire item collection. Consequently, though it is possible to initiate a scan of the whole classification file, a teacher must go to some trouble. This compromise between meeting user needs and discouraging use of unnecessary computer time has so far proven satisfactory.

The item file itself consists of 80-character card images, where each image contains one text line of a printed item and a unique identification number. No attempt was made to compress text by coding blanks. This file serves as the master file of items; there is no duplicate file of punched cards. When cards are desired to aid in changing item text, they are punched.

The item file is normally accessed differently for file maintenance than for test construction. Items are retrieved from the item file for tests by direct access, but the basic item file maintenance run is a sort-merge procedure which rebuilds all of the item collection-dependent files. Item additions, deletions, and substantial modifications may be accumulated and this run executed infrequently. Between such runs, it is possible to prevent specific items from appearing on tests and to change specific item cards in the file by direct access.

Computer Runs

Like those for item file maintenance, most computer runs were designed for use by coordinators. The principal runs available to coordinators are listed in Table III. A few other runs are for the programmer during system maintenance or when adding new item collections. The two primary runs, executed daily, are those which service teachers: one produces tests and the other handles scoring. Test production will be discussed here; scoring will not.

Table III. Principal Runs for Coordinators

Run Name	Function
Time-Out Cancellation	Remove old tests from the active list
Print Teacher File	List the teacher file and active list

Run Name	Function
Teacher File Maintenance	Add, delete, or modify teacher identification data
Print Activity Report	Produce report from system statistics file and reset counters if desired
Print Item Statistics File	List item statistics
Print Item Statistics Report	Identify items having statistics which exceed specified thresholds
Print Classification File	List item attributes
Item File Maintenance	Add, delete, and replace items, and reset an item's statistics if desired
Item Stop and Change	Tag item so that it will not be available to teachers, or replace item characteristics or text line

Test Production

There are two strategies which might be employed for batch retrieval of items: One is to publish a catalog of all items in the collection, from which the teacher selects those he wants; the other is to have the computer select items from the collection according to attributes specified by the teacher. When a large question data bank is involved, it is impractical for the teacher to deal directly with the questions. Furthermore, if, in addition, new items are continually being added and old ones revised, providing teachers with a relevant catalog of items becomes a problem. It would be hundreds of pages long, and the publishing cost would be compounded by the need for frequent revisions to account for changes. For these reasons, CTSS relies instead upon the computer's ability to retrieve by attribute, while still reserving to the teacher his right of final choice.

However, the approach chosen results in another problem. When selection by attribute is used to locate entries in a large data bank, the difference between the quantity desired and the quantity available must be resolved. This problem is easily handled in a conversational retrieval system, because the user can specify attributes and immediately learn how many items are available. If there were more than he wanted, he could tighten up the specifications and inquire again; if there were less, he could loosen them. In a batch retrieval system, some alternate

means needs to be employed to insure that the user is neither flooded with eligible items nor receives too few. In CTSS, random selection achieves the former and automatic specification relaxation the latter.

If, for example, a request block specifies five questions having certain characteristics and there are 100 that satisfy the criteria, five are picked at random from the 100. CTSS does this by partitioning the 100 questions, ordered by category number, into five groups of 20 items each. One question is selected at random from each group. The stratified sampling prevents too many items from being occasionally picked from a single category when selection ranges over several subordinate categories. There are, of course, many alternative ways to reduce the number of eligible items, but random selection has proven adequate.

If fewer items are found that match a request block's specifications than are requested, some specifications will be relaxed in an attempt to meet the quantity objective. This is consistent with the observation that teachers prefer to receive items, even though they do not meet all criteria specified. In the prototype, behavior level, if specified, is ignored first; then, any assigned difficulty level specification is disregarded. No other type of teacher specification is relaxed.

When more than one request block is used on a test-request form, each is treated separately for item selection purposes. However, no item is included in a single test more than once. In addition, the identifications of items deleted from a test during modification are stored in the active list. Such items are considered ineligible for subsequent generations of the test unless specifically requested.

The test production run consists of a series of programs, each of which operates on a batch of test requests and runs to completion prior to the next program's execution. This run handles both initial requests and requests for modifications to existing tests. Considering only requests for new tests, the primary functions of each program are summarized in the steps below:

1. Convert the format of optically scanned test-request forms to one more useful during program debugging and maintenance

2. Sort test requests by item collection (so that all item files need not be accessible simultaneously)

3. Edit test requests for teacher input errors and allocate space in the active list and version file

4. Select items to appear on tests by referencing the classification file; store identification of items selected in the active list and scrambling keys in the version file

5. Update the system statistics file

6. Prepare tests for printing by retrieving text from the item file

7. Print summary of this run

8. Print tests on paper (spooled)

9. Print tests which are to appear on reproduction masters (spooled)

Backup and Recovery

CTSS has an extensive set of backup and recovery procedures built in to reduce the impact of human or machine errors related to processing. Because all data needed for runs is retained in files, backup procedures need only be able to restore these files. Prior to executing the test production, scoring, teacher file maintenance, and time-out cancellation runs, the teacher file, active list, and version file are automatically copied. If the run does not go to completion, these files are restored. Because the most recent state of the quasi-random number generator used to select items and scramble tests is stored in the active list header, a test production run can be reproduced from restored files. The system statistics file is copied periodically. Every time item file maintenance is run, a copy of the item classification file, the item file, and the item statistics file is automatically made.

CONCLUSION

Probably the most significant systems learning that occurred during prototype operation has been in the support area. The fact that a good deal of attention was directed toward reducing teachers' chores turned out to increase the coordinators' work substantially. For instance, although teachers are encouraged to offer new items and suggestions for improvements, they are not expected to revise

questions. Likewise, teachers are not required to submit their names and locations with requests. Such system-provided services resulted in the presence of additional files and more support work for coordinators. The coordinators' activities have required more computer assistance than had been anticipated during system design. In fact, most of the functions added after CTSS was installed were to aid coordinators--either to diagnose teacher difficulties or to maintain files.

The CTSS prototype is available to other educational institutions; programs, documentation, and some of the existing item collections may be obtained from the Los Angeles City Unified School District. Several other institutions have installed CTSS and more data banks of questions are becoming available. Also, systems of a similar nature have independently emerged at various other locations, chiefly in institutions of higher learning. One can infer from the success of CTSS and from the developing interest elsewhere that the use of computers for banking questions and generating tests and exercises is an embryonic application area which will continue to grow.

Furthermore, test generation is a natural component of more sophisticated computer-assisted instructional approaches. A few of the existing automated test construction activities are, in fact, parts of larger computer-managed instruction systems. These more extensive systems usually include pedagogical decision-making elements, such as diagnosis of learner difficulties and prescription of assignments. Some of them enable students to proceed through large units of instructional material independently of each other. Those who wish to begin with a small, simple system and grow toward a comprehensive system may find test construction a convenient starting point, since it can stand alone under teacher control as well as fit into an integrated computer-managed instruction system at a later time.

last page