THE UNIVERSITY OF MANITOBA

A COMPUTER AIDED INSTRUCTION SYSTEM

BY

GINO BRAHA

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

WINNIPEG, MANITOBA

May 1972.

ABSTRACT

A description is given of the philosophy, design concepts, and implementation of an experimental computer aided instruction system, for automated teaching in a real-time environment. A short review of work on CAI is included.

ACKNOWLEDGEMENTS

I wish to express my thanks and appreciation to:

Dr. C. Abraham, my thesis supervisor, for his assistance and guidance in the development of TEACH system software.

Mr. Bill Reid, for helping me to generate a copy of the MUM production system and to make the necessary alterations to the terminal I/O interface program.

Dr. John C. Muzio and Professor Gavin Hoare for reading this thesis and providing the suggestions that led to subsequent improvements.

Miss Carol McQuarrie, secretary, for typing parts of this thesis.

TABLE OF CONTENTS

INTRODUCTION

The recent advances in computer hardware have been the catalyst to many fresh computer concepts. Along with the new concepts came the idea of using the computer to assist the educational media in the everyday chores of teaching. Thus evolved a major field of computer applications research, that of the philosophy, software development, and implementation of Computer Assisted Instruction (CAI) systems.

The organization of any one CAI system is such that it:

a. is an on-line time-sharing system capable of supporting a reasonable number of student/teacher terminals simultaneously in conversational mode.
b. supports a variety of audio/visual devices.
c. provides communication between user and the system in a natural language.
d. facilitates the generation and retrieval of tutorial information into and from data banks.

The differentiation between CAI systems lies in the philosophies attached to each of the systems, every one possessing its own approach as to what should be the relationship between teacher, student, and computer. The

philosophy behind the system influences the kind of program design, implementation, and choice of peripheral devices that are required in order to create a useful teaching environment for student or teacher.

Under a grant of the National Research Council awarded to Dr. C. Abraham, CAI reasearch at the University of Manitoba has led to the materialization of a computer aided instruction system which is called TEACH.

This thesis discusses the aims and reasons behind TEACH, its applications, and the hardware and software configurations. All topics are covered in five main divisions, the first two of which expand on the philosophy of TEACH. Chapter III comments on the internal software organization, with emphasis on the adaptation to a remote terminal monitor. This is followed by a user's manual and concluded with a description of one of the principal application programs in the system. Additional information on system use and maintainance is provided in the appendices.

All programs for the system are written in IBM/360 Assembler.

A REVIEW OF SOME CAI SYSTEMS

THE COMPUTER-AIDED TEACHING STUDY (CATS) AT NRC

The National Research Council of Canada is currently undergoing its second phase of a program of research in the development and evaluation of a CAI system (1).

Phase II uses a PDP-10 time-sharing computer with 32K words of memory and a 500K-word disk storage (1). Lesson material can be stored in English or French and generated from any one student terminal station. The system will act as a central facility providing for the generation of educational material on data banks that will be available to educators, NRC, researchers, and possibly schools.

The first phase began in 1967 with the introduction of a basic computer-aided teaching system (2), running on two small computers (one designed by NRC and the other, a PDP-8) connected to three time-shared teletypes. The central computer contained the course material, student records, and system control programs. The PDP-8 was employed as a teleprocessing buffer between the main computer and the terminals. The system became a source from which to obtain information and experience for the future design of more complex software and specialized input/output equipment in phase II. The system operated in 'tutorial' mode and

provided for linear and parallel flow in the presentation of lesson content (3,4). Commands were created to comprise a standard author language which facilitated preparation and editing of course material. A geography course was used to evaluate the system. Visual course contents were placed in a 'resource folder' and were manually accessed by students as instructed by the program stored in the computer. Each lesson could contain a maximum of 12,000 characters.

Other work being carried on at NRC, related to CAI, includes research on an x-y co-ordinate touch-sensitive position encoder for computer input (5). This would enable students using a CAI system to just point at information displayed on a cathode ray tube terminal in response to program-controlled queries. At the same time, research on computer detection of mispelled words entered at a terminal, has been considered (6).

CAI AT STANFORD UNIVERSITY

Research on CAI at Stanford University (7) began in 1963 with CAI control programs stored on a PDP-1 (32K) computer connected to six student stations within vicinity (100 feet) of the computer; each station composed of an IBM random-access optical-display device for microfilm display, a Philco cathode-ray tube, and a Westinghouse audio system for tape messages. Files were stored on two IBM 1301 disks.

Evaluation of the system proceeded with the introduction of an elementary mathematical logic course to fifth-graders. In the following years, the system was connected to dial-up terminals in elementary schools around the Stanford University area and, eventually, across the States.

Their research, recently, has been concentrated in the fields of grammar and semantics recognition, speech recognition, and the construction of mathematical models to test student learning and performance (8) . They make use of PDP-1, PDP-10, and PDP-8 computers with disk storage on two disk modules of an IBM 2314 disk unit. Direct storage access is handled by programs stored in the PDP-10 computer (fast memory cycle time). The PDP-8 acts as a teleprocessing buffer for some of the remote terminals. Text-handling programs are contained in the PDP-1 computer. Figure 1 displays the CAI system configuration at Stanford as of June, 1969.

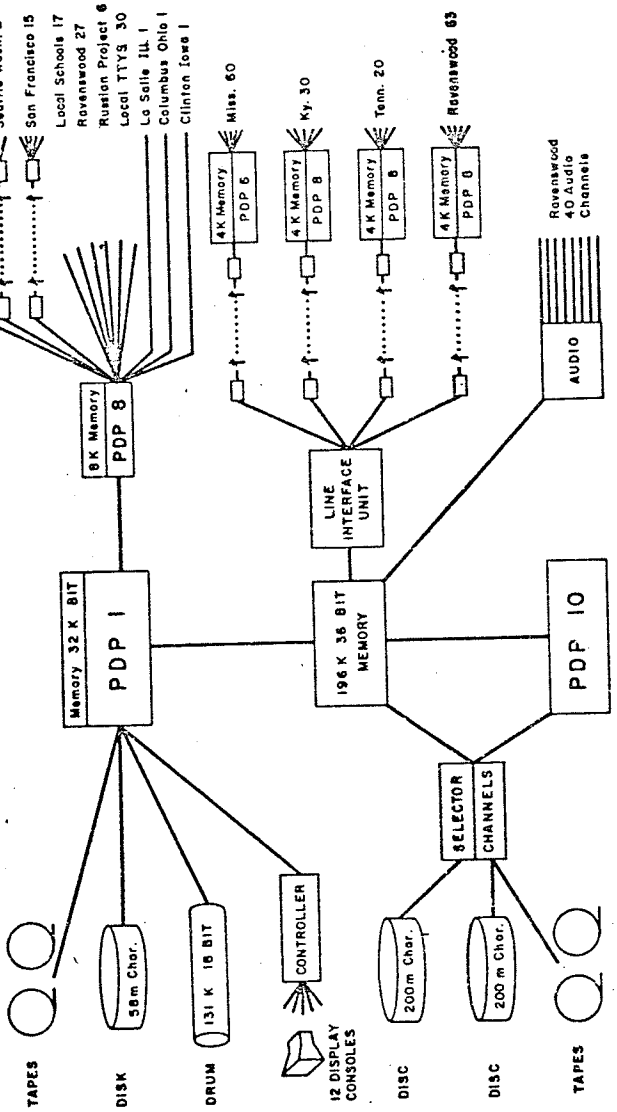


FIGURE 1. THE STANFORD CAI SYSTEM CONFIGURATION AS OF JUNE, 1969.

IBM COURSEWRITER III SYSTEM

The IBM Coursewriter III system (9) is designed to run on an IBM System/360 Computer. Students communicate with the Coursewriter III through IBM supported student/author stations. Course presentation takes the form of a show of slides, play of audio messages, and presentation of typewritten information. The terminal mix is irrelevant: CRTs can be used along with selectric typewriters or other terminals. Theoretically, the system is capable of handling up to 161 remote terminals. The student may respond to a question by using actual english sentences or words. Author commands resemble very much Assembler/360 instructions. These consist of an op code followed by an operand field. Most of the time, the operand field of the instruction is the message (text) to be output on the terminal or intended for the audio/visual devices. The NRC project uses a similar instruction structure.

Prior to the development of Coursewriter III, IBM was experimenting with their own CAI systems on 1050 and 7010 computers, using a CAI laboratory in German for evaluation of the system (10,11).

OTHER CAI SYSTEMS

Information has been compiled by the U.S. Office of Education on the CAI research projects funded by them and

going on throughout the United States. This list is elaborated on in (12).

CHAPTER I

THOUGHTS ON TEACH AND CAI

As a continuous involvement, educators are experimenting with new teaching methods - methods which, they hope, will improve the quality of the teaching and learning process. Therefore, it was natural that the computer should be considered as a tool for the teaching and learning process.

A mechanical brain having the characteristics of speed, low error frequency, and capable of remembering information, the computer could help the human being very much in the same way that a tutor may assist one of his pupils in times of incomprehension or need for references.

The computer is more than just a gadget to which one resorts when unable to find the proper solutions. Its greatest virtue is that it can handle several totally unrelated tasks simultaneously. To educators who always have their hands full, this last attribute is priceless.

The TEACH system was designed with the intention of providing, for the instructional media, a tool to assist in the individualized teaching of students.

For instance, the gifted student, usually ahead of his class, may find his school day very dull and dragging.

Frustration and tension may not be ruled out when he finds himself being 'educationally oppressed' by having to proceed at an equal pace with his fellow classmates. TEACH offers him an escape, so to speak. He may go to any student terminal and hold a conversation with TEACH on practically any pedagogical topic at his own level. He is capable of enriching his curriculum and might well experience a degree of motivation rather than frustration. The teacher is rid of a problem task and can have more time to spare for those problems facing other students.

On the other hand, the slow learner may need the extra practice and individualized instruction necessary to keep him in pace with the rest of his class. A private tutor might prove financially impossible. A terminal at school monitored by TEACH provides him with extra information on the topic of his choice, offers him some drill-and-practice questions, analyzes his answers, and provides helpful hints. Unlike the unfortunate event that many experience in school, the slow learner is not subject to ridicule if he offers a naive reply. Instead, he is able to build self-confidence in himself.

For those students who are disabled: confined to bed through illness, blindness, or physically afflicted in some other manner, TEACH allows them the benefit of classroom instruction even though they may not be present there. With such help, these unfortunates may become capable

and feel useful to society. This is not to mention that their morale would be much improved.

The business community may find TEACH helpful in a variety of ways - the first of which is retraining workers who might be considered obsolete owing to advancing technology: displacement by automation. A company might wish to bring executives up to date with current business matters and trends, or offer a course to their salesmen on the art of selling a product. Technical procedures, defining a sequence of steps to be followed in achieving a specific technical goal, could be stored in the system's memory files for access by repairmen, laboratory technicians, plant foremen, and so on.

TEACH is adaptable to automatic process control systems, in plants and factories. Proper interfacing of TEACH with plant equipment could provide computer control to machine cycles; for example, a dye beck in operation during the dyeing stage of a product manufacture.

This thought could further be expanded to include the medical profession who, besides teaching physiological topics to their students, could use TEACH to retain patient files, medical procedures as instructions to hospital staff, disease symptoms, and treatments.

Newly landed immigrants, as well as tourists, may be helped to integrate to the country's geography, customs, language, laws, federal system, and foods.

Municipalities may employ the system as a form of abridged directory providing information for plays, concerts, movies, operas, museums, landmarks, department stores, and tourist centers - the kind of data usually compiled in a 'what's doing?' brochure.

Universities, short of academic staff, may offer a wide range of extra courses which do not need lecturer intervention and which could not possibly be rendered otherwise. Since these courses would be 'lecturer-independent', no rigid time schedule need be followed, the course being available at any time of the day or night that the computer installation is functioning. Focus may be instituted on experimentation in course development.

A troubled individual may be provided with psychological or guidance counselling through the facilities of the system. It is somewhat probable that he may feel freer in communicating his thoughts to a machine while, in front of an examiner, he becomes self-conscious and reserved.

There exist a few more features and advantages to computerized teaching not familiar to TEACH so far but which could be incorporated into the system with further software development.

As an example, the teacher's regular administrative tasks of keeping a record on each student's progress and

background could be alleviated by letting the computer handle such routine matters. Analysis of student functioning and behaviour would be provided in a computer-generated report as requested by the teacher. This load taken off the teacher's mind allows him to concentrate more on his profession - that of being a teacher and not a record clerk. The benefits of such action are far greater in the long run, especially since the number of students and, accordingly, the demands made on teachers are increasing substantially every year.

An ideal computerized teaching system would be set up in a national or international centrally located installation, able to process all requests from educational institutions twenty-four hours a day. The facility would be available to any citizen who is willing to use it for reference or study. Members of the country's highly proficient group of scientists, educators, philosophers, doctors, and others would form boards who would provide the data and educational material to be stored in the computer's memory banks. In this way, an individual gaining access to the national system would have, at hand, some of the best compiled knowledge possible - at a moment's request. Terminals stationed in underprivileged and remote areas of the country could be linked to a CAI system and, thereby, allow the inhabitants to share in the knowledge of their more fortunate neighbours. Wide use of terminals in the home

should be encouraged.

A set-up of this sort would be a man/machine feedback type of structure where users could, in turn, inform the machine of ideas, problems, or suggestive improvements that they wish to express. The system can then serve as a type of communications network where various groups exchange priceless quantities of data. Data analysis and feedback features built into the system would provide valuable information for educators to brood over and come up with solutions to student learning problems.

Although the near-ideal system is foreseeable in the future, it is somewhat too ambitious and costly a project to undertake in a university environment. Mini teaching systems such as TEACH have or will develop. Newer concepts and approaches will mature. Basing ourselves on the knowledge gained from observing these model installations at work, a national network could evolve which would, in time, become a subsystem of a world-wide educational network.

Presently it is understood that, as in all other matters, the computerized teaching system is nowhere near perfection. Current models of computers lack the very important ability to think high thoughts, generate ideas, or make sophisticated decisions in the same way that man's mind functions. Most of all computers lack the very ability to be creative. We have yet to introduce the kind of software whereby the computer will recognize the student's answers to

contain the proper concepts but who, for one reason or another, happens to use the wrong words to express himself. The computer is unable to portray sincere emotions that are constantly in evidence in the classroom during student/teacher interaction and which play a very important part in growing up. This is not to say that the machine cannot be programmed to achieve some higher level of thinking and decision-making. On the contrary, the recent developments in artificial and machine intelligence and heuristic methods may prove the above statement closer to reality than we currently believe it to be. Undoubtedly, much of the future research on CAI will be directed toward language syntax and semantic recognition algorithms.

Psychological problems, some as yet unknown and derived from machine depersonalization and alienation factors, will originate from the introduction of a large number of computerized teaching systems. General acceptance of CAI by the masses will prove difficult, owing to humane inertia, resistance to changes. Teachers might believe their job security jeopardized, especially since the major structure of the educational system will, undoubtedly, undergo change to adapt to a new situation.

A defeatist attitude need not be taken on considering the psychological influences of CAI. A massive education program on the subject of CAI would have to be undertaken to familiarize the general public with the

potentials of computerized education. Installing CAI systems would have to proceed in stages to allow time for user adaptation to the new environment. There is no reason why, at later stages of development, such systems could not be able to carry on conversations with students on a personal level- even a first-name basis. This shall become highly feasible when current research on voice recognition patterns will eventually lead to oral communicative interaction between user and system. Stored lessons should consistently include words of encouragement intended for the student.

Some ideas, which are similar in context to those mentioned in this last section, have been expressed for CAI systems in the references: (1,13,14,15,16,17).

CHAPTER II

THE STUDENT/TEACHER STATION MONITORED BY TEACH

The TEACH system is capable of monitoring a number of dial-up student/teacher stations simultaneously, each station composed of:

(1) An IBM 2968 Model 11 Audio/Visual Control Unit

(2) An IBM 2740 Model 1 Communications Terminal

(3) A Kodak RA-960 Carousel Slide Projector

(4) A Uher Model 5501 Tape Recorder

(5) A Data Coupler

These do not constitute a limit in the type of equipment that could be supported by the TEACH system. Minor modifications to the channel programs of the terminal input/output support routines could provide for a variety of communications terminals including CRTs. Little change is needed to the application programs of the system to allow the addition of a Braille input/output device for use by the blind or other audio/visual device; for example, a videotape. The more devices that one attaches to the system, the more flexible TEACH becomes.

The number of stations is a function of the equipment available at the computer installation.

So far, the student/teacher station has the

terminal configuration displayed above. Contact with TEACH is established by dialing up the system and receiving a reply. The student or teacher communicates with TEACH through the keyboard of the 2740. Tutorial material is presented to the student in the form of typewritten text on the 2740, a show of slides on the Kodak projector, and/or the playing of a pre-recorded message on the Uher tape recorder. The system allows, as well, a user (student or teacher) to send at anytime messages to the computer operator's console. Therefore, a teacher sitting at the operator's console could answer any queries that might arise while students are 'signed-on' at the student/teacher stations.

When proper contact has been established with TEACH, the terminal configuration enters in one of two possible states; that is, either in 'create' mode or 'teaching' (drill-and-practice) mode.

In 'create' mode the teacher employs the terminal configuration to prepare, edit, and map the contents of a lesson for eventual future presentation to his pupils. The lesson is stored as a file on disk space reserved by the system. The lesson file is available to students until a request for the deletion of the file is issued by the instructor.

In 'teaching' mode, the student sitting at the terminal station is steered through the lesson material

while, concurrently, the system is analyzing the student replies and, basing itself on the latter, makes momentary decisions on which course path to follow next. Although several stations might exist simultaneously in 'teaching' mode, students at these terminals may have access to totally unrelated subjects.

When TEACH is no longer required by the user, the latter at the terminal station issues a 'sign-off' command.

LESSON ORGANIZATION

Lesson material, on the TEACH system, may assume the form of:

a. a straight lecture.

b. a true-false quiz.

c. a multiple-choice examination

d. a lecture followed by a quiz.

e. a fill-in-the-blanks session.

Lesson content is recorded on disk files, slides, and/or tape reels. Each lesson file on disk can retain a maximum of 74880 characters. Eighty slides may be contained in the Kodak circular slide tray. The slide trays are removeable and interchangeable. Therefore, there is virtually no limit to the number of slides that may be presented during the course of a lesson. The same holds true

for tape reels, where maximum content per reel is in the order of 1023 recorded messages - each three seconds long. Subject to a code inserted in the lesson during its creation, TEACH will inform the student when a change of slides tray or tape reel is due. The order of audio/visual presentation does not matter. Slides or tape messages may precede each other. However, both must appear before typewritten text.

A 'lesson map', defining the sequence of steps to follow and any remedial action to take during course presentation, is stored as part of the file containing the lesson. The purpose of the 'map' is to guide the student through the lecture or examination material. Further discussion on the concept of the 'map' in TEACH will appear in the next subdivision of this chapter.

The lecture may cover any topic that one so desires: a lesson in history, a quiz in mathematics, a true-false test on chemistry, or the philosophies of Plato and Aristotle. There exists no restriction on the type of material considered. It is as easy to include complex mathematical equations as it is to insert English text.

The actual lay-out and content of the lesson material is left entirely up to the teacher's discretion. It is suggested, however, that the lay-out be structured on a set pattern initially (Figure 2), especially when familiarizing oneself with the system. The lecture may start

off by generating an abstract for the material to be covered which includes a summary of objectives aimed at in the lesson and course. The student is, thus, able to orient himself to the topic under discussion. A continuous alternating sequence of instructional material, succeeded by questions on the content of the latter, follows the introduction. The concluding portion of the lesson may include an overall review of the subject matter covered in the lecture, a summary of the topics covered in the course to date, and a series of review questions.

Students' replies to questions asked in the lesson can only be in the form of numeric digits: one to five. For this reason, all questions must be accompanied by all correct, probable, and wrong answer(s), each answer having associated with it a numeric value (one to five). In other words, questions must be structured as if they were multiple-choice.

Should the student's reply be erroneous, the lesson could supply him with extra explanation or hints and ask him to try again; or else, branch to an altogether different portion of the lesson and continue on an alternate path to possibly clarify the mistake(s) made.

A SUGGESTION FOR LESSON MATERIAL ORGANIZATION

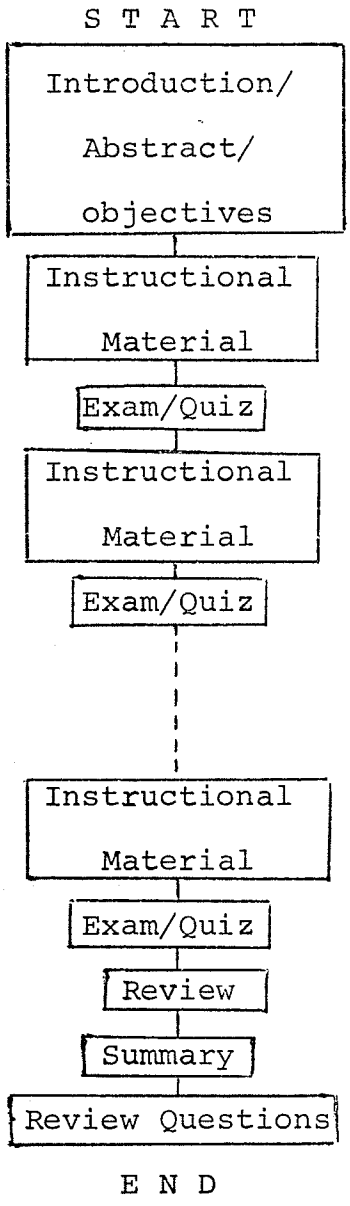


Figure 2

The following is a sample script of the student/TEACH dialogue on the subject of the Hydrogen atom. All commands that would regularly be issued to the system are left out on purpose for sake of simplicity in reading the dialogue.

Slide 1 : LESSON #10. SUBJECT: THE HYDROGEN ATOM.

Tape Msg #1 : "The purpose of this lesson is to study the characteristic properties of the simplest of all elements - hydrogen, the first element in the periodic table."

Text : In order to study and understand the characteristic properties of hydrogen as a gas, we must examine the structure of the hydrogen atom. One way is to consider the emission spectrum of hydrogen photographed when radiation given off by the element is passed though a spectroscopre. Observe the contents of the next slide.

Slide 5: Emission Spectrum

x= 6562.8 (wavelengths)

y= 4861.3

z= 4340.5

Slide 21 : Balmer's equation: $1/L=v=R\ (0.25-(1/(nxn)))$

n= 3,4,5,.....

Tape Msg #16 : "where L is the wavelength, v is the wave

number of the spectral line, and R is Rydbergh's constant."

Text : The equation always will be positive for values of n greater than 2.

1. True 2. False

System: ANS.

Student reply: ?1

System: CR (for correct reply)

Text: Now assume that R can be approximated by the value of 10 to the power of 5. What is the value of 1/L for n=3 in terms of 10 to the power 5.

1. 0.250
2. 0.138889
3. 0.13333
4. 1.33333
5. None of the above.

System: ANS.

Student reply: ?3

System: WR (for incorrect reply)

Second try: ?1

Text : Hint: A second attempt may be made. Evaluate the contents inside the brackets first.

Student reply: ?2

System: CR

Text: Hydrogen is the _____ element in the

periodic table.

1. 1 st 2. 6 th

3. 9 th 4. 96 th

System: ANS

Student reply : ?1

System: CR

THE CONCEPT OF THE 'LESSON MAP'

In a classroom environment, a perpetual occurence of decision-making is in evidence. Examples of the kind of decisions taken by a teacher might be:

(1) what sort of information to expose to the class,

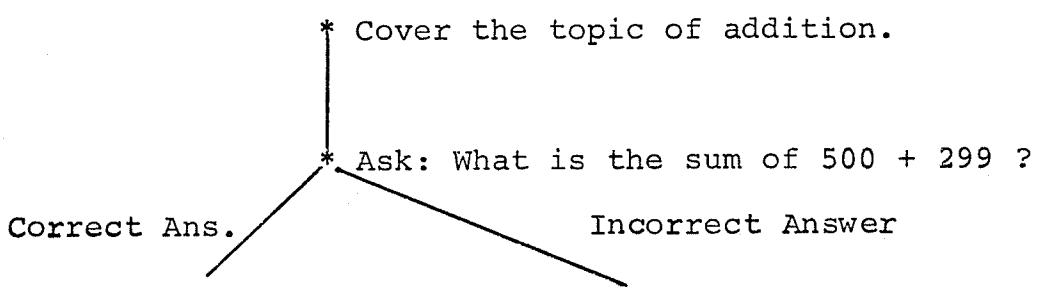
(2) how to process an erroneous reply or misconception,

(3) how to handle a correct reply,

(4) whether or not to offer more data to the class to guide them along on a specific topic, and,

(5) how to respond to an individual's query.

The decisions are subjected to the teacher's experience and professionalism. A closer analysis of the decision-making process witnessed in the classroom will reveal a definite constitution to the process - that of a tree or branching structure (Figure 3).



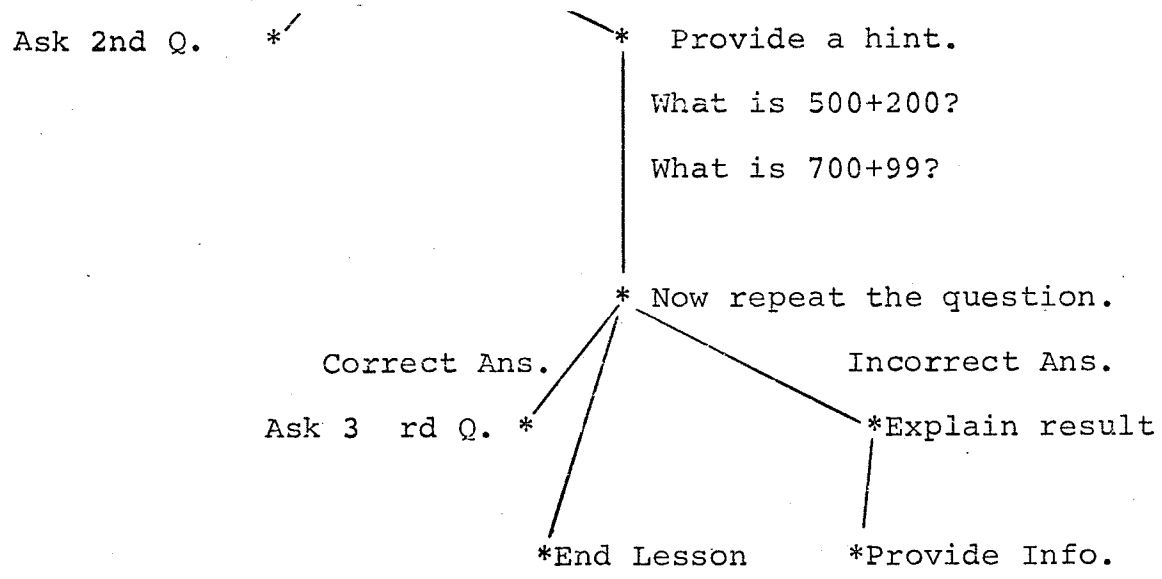
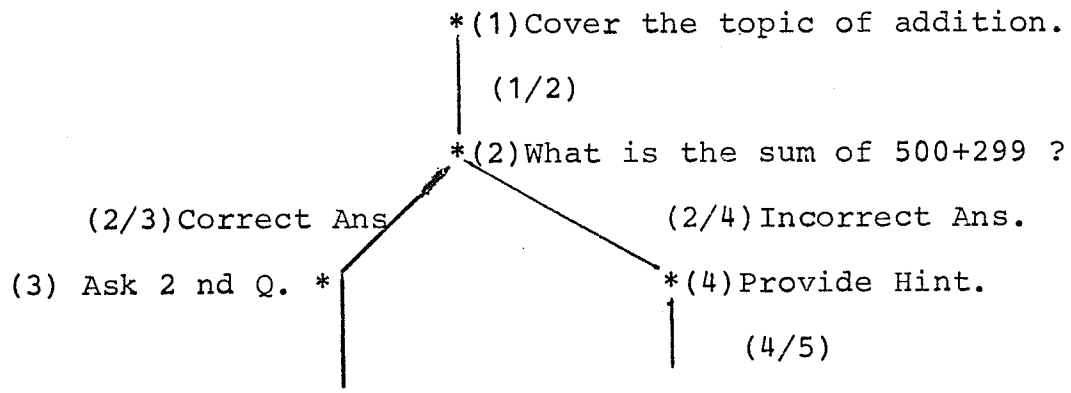


Figure 3

One may note that the tree-structure possesses a node at every point where an important action is taken by the teacher. Suppose the nodes are numbered consecutively, starting from the top of the structure. Moreover, we represent the path between two nodes by 'starting node - slash - finishing node'. We, therefore, obtain this extended version of the tree-structure:



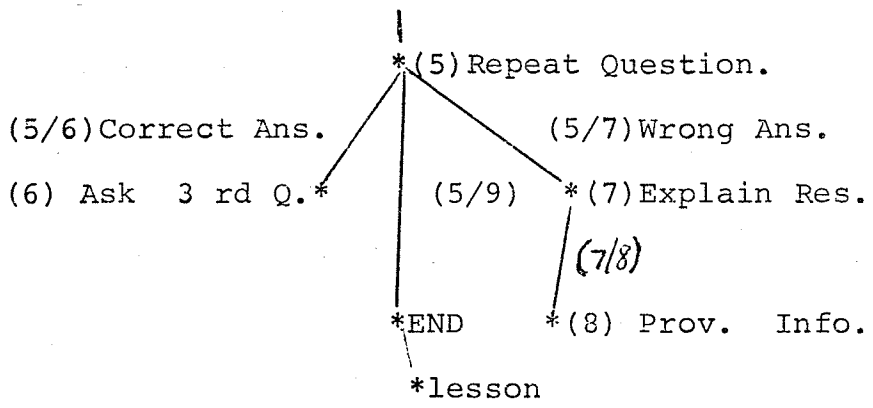


Figure 4

And, if we remove the text from the tree-structure, we have:

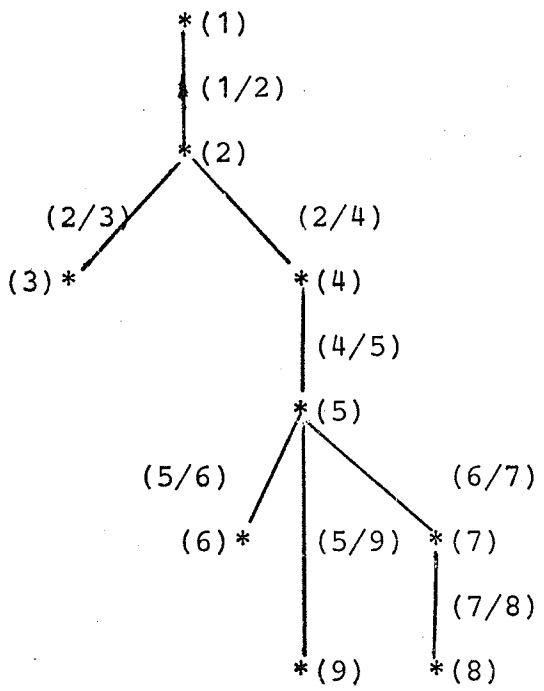


Figure 5

We may represent the above tree-structure, by the following table:

Node#	TrueCond.	FalseCond.	NoCond.
1	-	-	(1/2)
2	(2/3)	(2/4)	-
3	-	-	-
4	-	-	(4/5)
5	(5/6)	(5/7)	(5/9)
6	-	-	-
7	-	-	(7/8)
8	-	-	-

Table 1

where '-' is reserved for the insertion of other branches of the tree structure at nodes 1, 2, 3, 4, etc... One reads the table as follows: If a true condition exists as a consequence to Node 2, then proceed to node 7. However, if a false condition evolves from an action at node 5, then proceed to node 6, and so on. We remark that the Node# and the number preceeding the slash(es) in each row of the table are identical. For simplicity sake, we remove those and we find ourselves with a table resembling a two-dimensional matrix.

Node#	TrueCond.	FalseCond.	NoCond.
1	-	-	2
2	3	4	-
3	-	-	-
4	-	-	5
5	6	7	9
6	-	-	-
7	-	-	8
8	-	-	-

Table 2

In the TEACH system, we designed a facility which permits the teacher to simulate his own decision-making process. The development of this facility is highly based on the tree-structure pattern discussed and provides for multi-path alternatives to the presentation of a concept. The simulation takes the form of numeric data mapped onto a matrix table, similar in structure to the one above. It is this matrix which is referred to as the 'lesson map'.

The data entered in the matrix defines:

(1) the start and end (the nodes) of each question contained in the lesson.

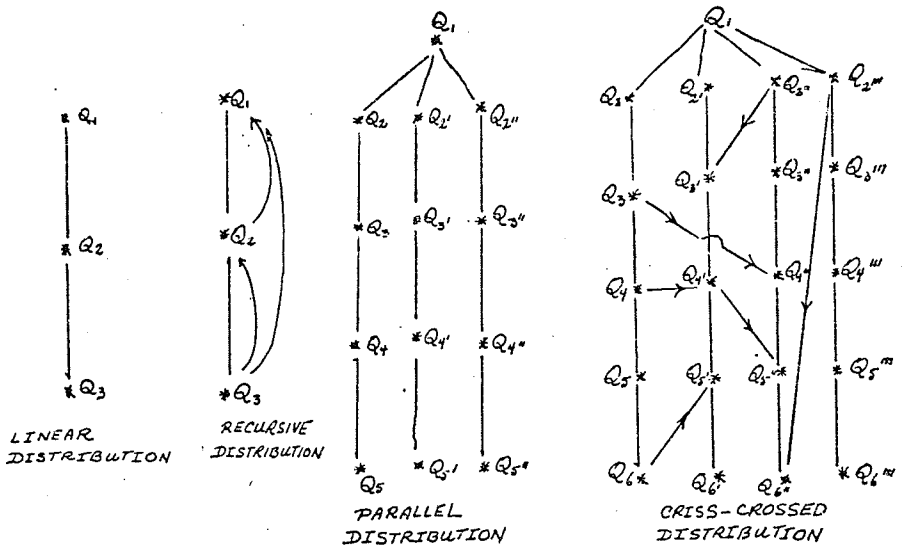
(2) the start and end (the nodes) of straight information for contemplation by the students,

(3) the correct reply for each question,

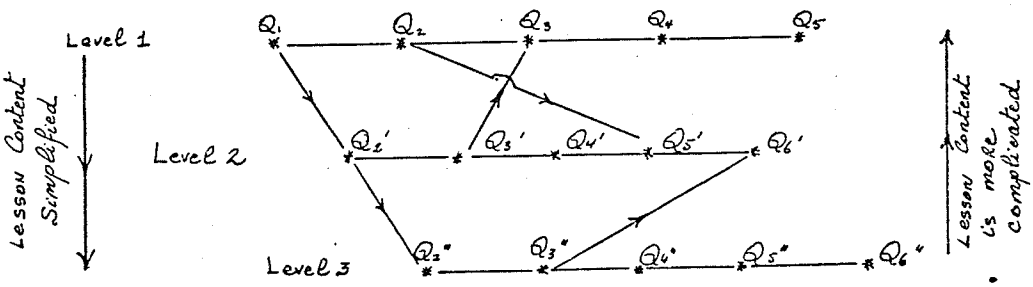
(4) the lesson material to be outputted if an erroneous reply is beheld, as well as,

(5) the slides and tape messages to be offered during audio/visual interaction.

The size of the matrix varies with the application program in use. The matrix allows for numerous different paths to be taken during the presentation of lesson material. The paths may be linear, parallel, criss-crossed, or recursive. The parallel and criss-cross paths are the most complex. A diagramatical representation of the paths is shown in Figure 6. The implication of these paths is that a bright student may skip over much of the lesson content that is normally reserved for the sake of clarification to the slower learning student. Since varying rates of learning are possible, different paths may be defined which correlate with the rate of learning of the students. This is individualized teaching at its best.



A. VARIOUS PATH DISTRIBUTIONS FOR THE PRESENTATION OF LESSON MATERIAL AS MAY BE DEFINED IN THE LESSON MAP



B. A SUGGESTION FOR LESSON PATH ORGANIZATION (And defined in the Lesson Map)

Figure 6.

CHAPTER III

TEACH SOFTWARE IMPLEMENTATION

Currently, the TEACH system runs under control of a subset of the Manitoba University Monitor (MUM) (18,19) in 40K of core on an IBM/360 Model 65 computer, having access to a 2703 teleprocessing control unit and 2314 disk pack. MUM is an on-line time-sharing system for users at remote terminals. This subset of MUM, somewhat altered to suit the purposes of TEACH, is a system in itself, totally independent from the production copy of MUM. Both the production MUM and the TEACH system can run concurrently without interfering with each other's functions. The TEACH system configuration is shown in Figure 7.

Four major software entities make up the vital components of the system.

The first of these is a set of monitoring modules (hereafter called the Monitor), extracted from MUM and fundamental to the operation of TEACH. These are assembled and linkedited into load modules which reside in a partitioned data set when not executing in core. The Monitor assumes control/service and request handling functions for the system. Control and service routines have, as principal task, the regulation

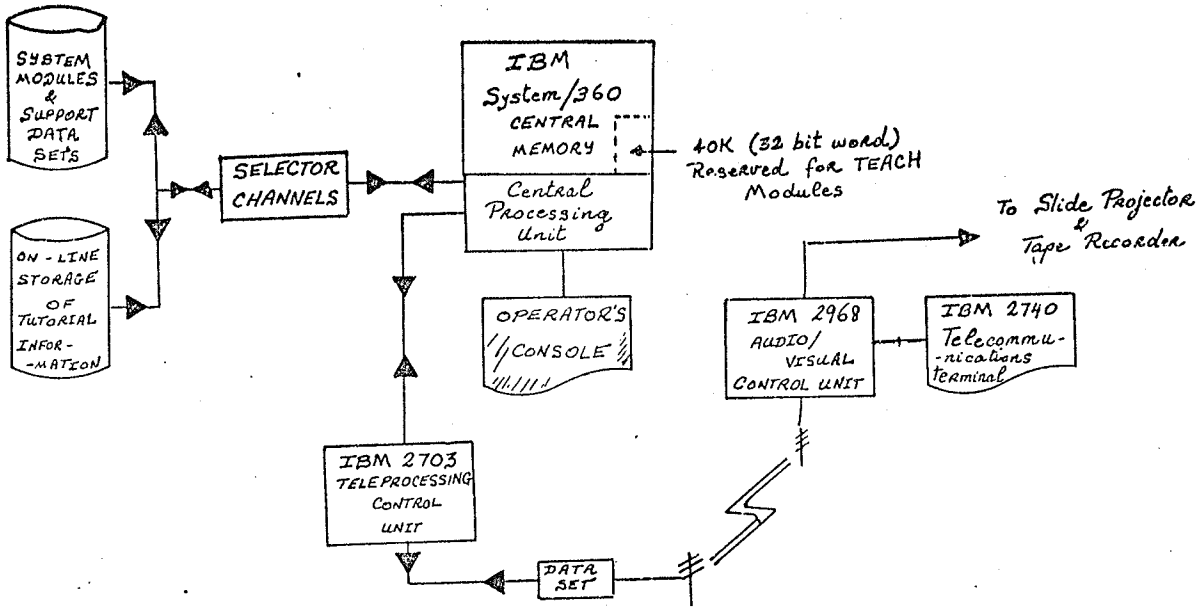


FIGURE 7. TEACH CAI SYSTEM CONFIGURATION

(As of April 1972)

and control of information travelling to and from the managing routines and the request handlers of the Monitor, as well as the information passed back and forth from the terminals. An example of a managing routine is one which has the primary responsibility of calling on the next application program to be executed. An application program might make a request to a request-handling routine of the Monitor for the transmission of a message to a terminal. The request handler processes the request and calls upon a service routine.

The remaining three components of the system reside, as data sets, on an IBM 2314 disk pack.

The first of these permanent data sets, an aggregate of contiguous formatted disk tracks, is reserved for storing files containing lesson material. The files are retrievable at subsequent sessions. An application program, requiring space to contain the lesson, reserves a definite number of disk tracks defined at assembly time. For each user account number a maximum of five groups of sixteen tracks each or eighty tracks in all are reserved to provide storage for up to twelve lesson files. The files can be made private or public. In the private case, the lesson file is accessible solely to that user account number under which the file was created. On the other hand, the public file is retrievable to any user account number.

All accounting information is kept on the second

data set. Each account number has a record of fixed length associated with it. The record contains such pertinent information as file name, creation date,and tracks allocated to that account number. The position of the record is numerically equivalent to the account number.

Processing of all user requests issued at the terminals is handled by the application programs. These are programs which solve user problems and provide the interface for interaction between system and teacher or student. For instance, one application program may simulate a desk calculator, another may provide for the creation and maintainance of a lesson. The application programs are mutually exclusive of the Monitor. They comprise the third data set known as the roll library.

The Monitor sets aside a roll-in/roll-out area in core, limited in size to 7294 bytes, in which it is constantly rolling in and out application programs as required. Therefore, the size of each application program is limited by the dimension of the roll-in/roll-out area (equivalent to the length of one track on the 2314 disk pack).

The maximum size that an application program may assume, namely 7294 bytes, is usually insufficient to contain all the code and data areas for several functions that one would like to see the application program perform. In order to overcome this handicap, tables and data areas of

the program are superimposed over assembler code that will not be used more that once. This saves space which might be used more profitably to increase the functions of the application program and, thereby, increase its potential and flexibility.

We are only able to do this, though, because of the manner in which the Monitor manages application programs. When an application program is requested by a user, a copy of the program is first placed in a scratch area, then rolled into core at which time it becomes active and available to the user. The original application program is not altered in any way and remains a resident of the roll library. When a new lesson file is created, the 'copy' becomes an integral part of the file. Access of the file at a later stage brings in this 'copy' into core for execution. Once the lesson file is generated, the assembler code which processed the file creation is no longer required. We can use the space occupied by this code to store other information relevant to the file and program copy.

Initially the first program rolled-in to establish communication with the user resides on the first track of the roll library. This program requests the user to sign on and to name the application program he desires. The application program requested is rolled-in next.

ADAPTATION TO MUM

Originally, the input/output interface routines of the Monitor provided for two-way communication between the system and such devices as an IBM 2741, 1050, or TWX telecommunication terminals. The interface routines generated proper channel programs for I/O to these devices. However, as MUM stood, it was unable to support the 2740/2968 TEACH terminal combination. Changes in the channel program, extension of buffers to contain the channel program, and alternatives to sections of code were effected to provide the TEACH system with a means of communication with the 2968 control unit (20) and 2740 terminal. In doing so, support for the original devices such as the TWX has been temporarily removed. Duplication of code could re-establish proper telecommunication support for all terminals.

The I/O interface program checks for an End-of-Transmission control character (equivalent to pressing the RETURN key on a 2741 or the EOT key on a 2740) at the end of user messages entered on the terminal. However, the replies from the 2968 control unit, which are not under user influence, contain an End-of-Block character as the last character transmitted to the system. Therefore, the I/O interface program needed some extension to include an attempt at locating an End-of-Block character, if it

failed to identify an End-of-Transmission character.

The 2740 terminal performs longitudinal redundancy checking on all messages received by it. The 2968 checks for proper length and odd-parity in the characters of the message received. If, owing to poor reception, an error condition is detected, then the terminal station responds with a negative reply. Ignoring this reply could produce internal complications in the sequence co-ordination of teaching events while the system is in execution. A routine was inserted in the I/O interface program to detect this specific negative condition and to resubmit the message, if necessary.

MUM made it possible for on-line job submission of programs to System/360. No on-line job submission would ever be required while signed on to the TEACH system. For this reason, the job submission capability of MUM has been removed by discarding the DD card that defined the unit address of the HASP internal reader, from the JCL required to load and execute the system. Had this not been done, the TEACH system and production MUM would compete for access to the same internal reader making simultaneous execution of both systems impossible.

One of the Monitor modules is a configuration deck which provides a link amongst all Monitor programs, supplying needed parameters and control blocks to the system, and containing macros which generate

telecommunication ports into the system for the terminal stations attached to the latter. This configuration deck contains ale the BTAM translation tables for conversion from EBCDIC to device-dependent internal character representation. A minor modification was made to the translation table of characters sent down the transmission line to the 2740. This consisted of SUPERZAP'ing some unprintable control characters required in the i/o support of the 2968 control unit. Otherwise, any unprintable control characters would have been translated to hex '88' which would have consistently resulted in an error condition being detected at the receiving station, making useless any attempt at sending instructions to the 2968 control unit.

MUM uses a Supervisor Call instruction to communicate with jobs running in the batch partitions. Since this facility is needed in some instances in the TEACH system, the SVC parameter of the $CVT macro, included in the configuration deck, has been set to 250 for the SVC call (21).

At the same time, macros generating ports for support of three 2740/41's, one TWX, one 2260, and one 1050 terminal were removed. Remaining macros were $OPER and $TERM, providing support for the operator's console and 2740/2968 terminal combination.

I/O SUPPORT FOR THE 2740/2968

The following sequence of steps are performed by the channel program to provide I/O support for the 2740/2968 terminal combination.

A Write/Read request is considered:

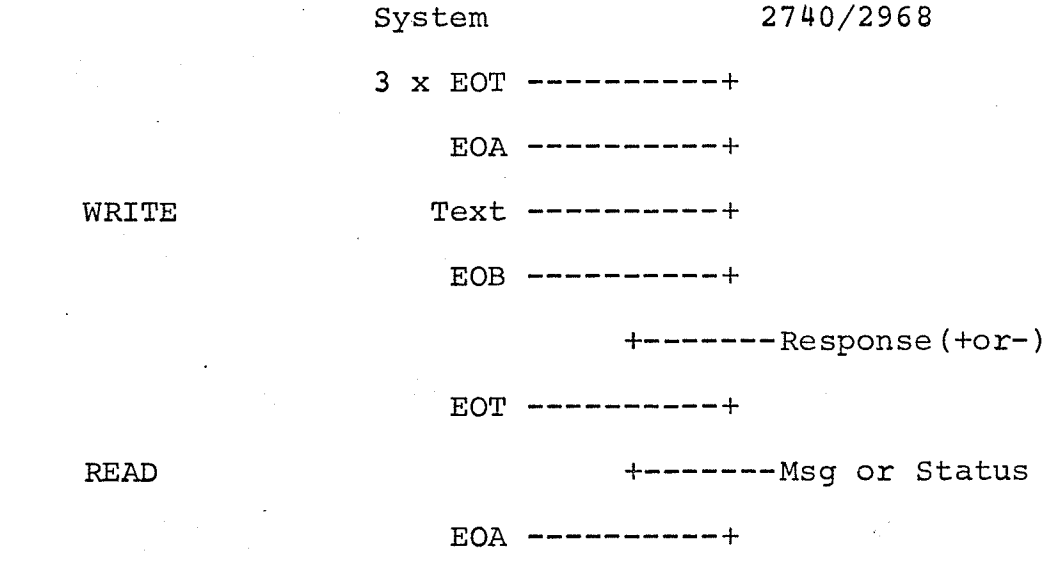
1. At first three EOT (End-of-Transmission) control characters are sent down the line to the terminal in order to reset the line for transmission.
2. An EOA (End-of-Address) control character is issued immediately following the EOTs. This EOA character forces the terminal station to enter in text mode in order to receive the message or text sent from the system.
3. The actual message that is to appear on the terminal or to be interpreted as a command to the 2968 control unit is then sent down the line.
4. When transmission of the message is terminated, an EOB (End-of-Block) character is sent. The EOB forces record checking on the message sent and makes the system await a reply from the terminal station.
5. If no error was detected in the message sent, the terminal station responds positively. A negative response forces TEACH to resubmit the message.
6. The positive reply is followed by an EOT to reset the line. This EOT character also causes the 2968

control unit to execute any command issued to it.

7. A prepare command in the channel program puts the receiving station, in this case the system, in text mode. At this point the user can enter on the terminal a message intended for the system or the 2968 control unit might provide a status reply on the condition of the audio/visual devices if instructed to do so.

8. The channel program is terminated with an EOA control character. For continuous Writes or Reads, the EOA control character makes it possible to skip steps 1 and 2 providing for quicker I/O interaction.

Diagramatically, the channel program looks as follows:



CHAPTER IV

APPLICATION PROGRAMS

All the application programs of the TEACH system together provide on-line facilities by which teachers can:

a. create new files, on disk storage, to contain lesson material on any chosen topic.

b. occasionally update, edit, and review the generated information.

c. define and update a sequence map for the lesson content (Included in the sequence map is the pattern to be followed in presenting lesson material on the audio/visual devices).

d. set the order of presentation of the lesson material:- whether slides are to be shown first, followed by a tape message, and concluded with typewritten text or if a tape message should precede slide display.

e. form modular problem banks to contain a mixture of examination questions and problems.

f. insert type-setting commands for proper indentation and underlining of text.

g. obtain a record of student progress for the duration of the teaching session.

h. construct a list of passwords for students so that they alone may access a particular lesson.

i. render inaccessible privileged instructions that manipulate lesson content.

j. declare lesson files to be for private or public use.

k. display lesson files status.

l. end communication with the lesson files.

m. abandon or delete a lesson file when its use has been exhausted.

The student, on the other hand, may use the application programs to:

a. access a particular lesson file.

b. start the instructional session or draw individual examination questions from a problem bank.

c. obtain a score based on his performance during the teaching session.

d. end communication with the lesson file.

In addition, there is available an application program that simulates a desk calculator (22) ,producing computed results in double-precision mode and which may also be used to evaluate several basic mathematical functions for any value x. This program may be called at any time during the teaching session if a student believes that he needs the help of a desk calculator to answer a question.

No single application program for the TEACH system

could, by itself, provide for all the facilities listed above. The very limitation in the maximum size allowed for an application program residing on the roll library, makes this an impossibility. In view of this, the functions performed for the user are forcibly shared amongst four application programs: TEACH, TEACH1, QBANK, and INDEX.

All of these application programs contain some identical servicing procedures (e.g. a lesson file may be generated from any one of the programs). However, each program contains functions that are unique to itself - not attempted by an associate. As an example, the TEACH1 application program is the only program which supports audio/visual instructional aids but can only process a very limited repertoire of editing commnads. TEACH, on the other hand, has a wide range of editing commands but cannot handle a/v equipment. Space usually reserved for editing commands had to be sacrificed, in TEACH1, in order to provide room for the managing routines of the audio/visual devices. The purpose of the application programs shall become clearer as the attributes of each one are described.

The TEACH application program yields for easy input of lesson material and facilitates correction of information (text) previously produced. A maximum of 55 question-notions may be contained in a lesson file created under TEACH. As already stated, no allowance is made for a/v equipment.

The TEACH1 application program provides for the

inclusion of audio/visual aids. Editing commands are made available for changes to the contents of the lesson map and the student password buffer. Student performance, on a lesson, is tabulated. A file may be created under TEACH1 but provisions for on-line insertion of lesson material are not available. The lesson file must be submitted in batch mode, with the help of a TEACH system utility program. In view of the a/v support built into this program, the lesson material capacity is of a higher order than that known to the other three application programs (information could now be stored on both the slide projector and tape recorder as well as on disk). The lesson map may contain data for a maximum of 255 question-notions. Text editing and type-setting commands are non-existent.

The QBANK application program allows the generation of problem banks to contain drill- nd-practice examination questions. With each problem in the bank is associated an entry into a table or map (with a variable number of entries) which represents the correct reply for that particular problem. The correct reply may be a string of any character combination, including numeric digits for multiple-choice answers.

The INDEX application program (23) allows users to generate small data banks. Each data bank is composed of an unlimited number of lines of textual information. Along with the number of lines of textual information is associated an

'index' word or 'key' formed from a string of characters. Retrieval of the textual information corresponding to the index word is made possible by entering the related key. The program INDEX may be used to create a dictionary for any language (be it a natural or computer language), a table of constants, scientific formulas, course outline, keywords for the theme of a course, amd may be used as a teaching aid.

The EXAM application program is currently under design and would be added to the system at a later date. The purpose of EXAM is similar to QBANK in that it provides the facility by which a teacher is able to insert examination questions and problems onto a data bank. However, the differences lie in that the examination questions could be offered to the student on a totally random or sequential access basis. The order would not be pre-determined by the author of the questions. At the same time, a sophisticated system of grading results of the examination would be incorporated into EXAM - a property that the QBANK program does not possess.

Which of these application programs is best suited for the kind of educational material that one may have in mind, is dependent on the instructor and what the instructor wants to do with the material, where he wants to store it, and how he wants it presented.

SIGNING ON TO THE SYSTEM

Contact with the TEACH system is achieved in a similar manner as with MUM; that is, by dialing the phone number associated with the port of the system. When proper communication has been established, the Monitor will respond with the following message on the terminal:

ENTER ACC'T #, PRG NAME

The teacher/student is requested to sign on by entering his assigned account number with the accompanying password, if any, and the name of the application program he desires to use.

e.g., 30.pass teach1

The above example displays the user account number '30' followed by a desire to obtain a copy of the TEACH1 application program. The password, represented by 'pass' in the above example, is optional. Nevertheless, it is the responsibility of the owner of the account number to attach a password in order to prevent illegal use of his account number. When the account number is initially assigned to the user, there exists no password. To insert or change a password, the following request must be made to the Monitor:

30 ??ch tsys (no password before)

30.pass ??ch tsys (password initially)

Thus, 'tsys' becomes the new password.

Signing on to the system may be done implicitly or

explicitly. By signing on implicitly a user must enter his account number and password everytime he requests a new application program or lesson file. In order to sign on explicitly, the user must precede his account number by the word "on". In this case the user will not be required to enter his account number and password everytime he makes a new request for programs or files. To sign off after termination of application program use, the user types in "off".

Assuming proper signing on procedure is achieved, the desired application program will be brought into core and execution will begin. These messages will appear on the terminal, depending on the application program that is called:

PRG. EDIT IN CONTROL (for EDIT)

PRG. ETEXT IN CONTROL (for ETEXT)

PRG. TEACH IN CONTROL (for TEACH)

PRG. QBANK IN CONTROL (for QBANK)

PRG. INDEX IN CONTROL (for INDEX)

PRG. TABLE IN CONTROL (for CALC)

TEACH SYSTEM IN CONTROL * ENTER REQUEST (for TEACH1)

At this point, the teacher/student is allowed to submit valid commands to the application programs for lesson creation, service, or control.

It is not necessary to await a reply from the application programs that they are ready to accept and

process commands. The first command may be inserted immediately following the signing on procedure:

e.g. on 30.tsys teach new lesson.file

This becomes time beneficial to both the user and the system.

TEACH1 USER COMMANDS

TEACH1 user commands belong to one of three sets. The first set of commands consists of those lesson file access instructions which may be used to:

a. create a new file.

b. access an old file.

c. change the key on a file.

d. declare the status of a file (pub. or priv.)

e. display file status.

f. end communication with the TEACH1 program.

Instructions which generate information to go with the lesson file, belong to the second set. They are mainly used to:

a. generate, update, and review the lesson map.

b. produce a list of student passwords.

c. obtain a grade for student performance.

d. start the teaching session on its way.

e. delete a lesson file.

f. end communication with the lesson file.

These instructions can only be used while the terminal station is in 'create' mode.

Commands of the third set are valid only when the terminal station enters 'teaching' mode and can be used to:

a. present the next slide or tape message.

b. repeat a question.

c. end the 'teaching' session.

Every command need not be written out in full. In most cases, only the first letter of each instruction will suffice.

Whenever signing-off any lesson file or application program, the time elapsed while signed on to the program will be outputted on the terminal.

The First Set of Commands:

new filename.key:

A new lesson file is generated with a 'new' command. The file name tagged on to the lesson file is defined in the operand field of the instruction. A protection key may be concatenated to the file name in order to restrict access to the file.

The file name can be any string of up to six alphanumeric characters. The protection key may be any

string up to four alphanumeric characters. Exceeding any of these lengths will be ignored. Duplication of file names under the same user account number is not allowed. A limitation exists on the number of lesson files that may be generated under the same user account number.

old filename.key

A lesson file, generated previously and currently residing on disk storage, can be retrieved by the 'old' command. The name of the requested file and the protection key associated with it are entered in the operand field.

key filename.previouskey newkey

In order to change the protection key on any lesson file, a 'key' command needs to be issued. Included in the operand field of the instruction are:

a. The name of the file on which a change of key will take place.
b. the previous key name,
c. the new key name.

present:

All lesson files generated under a particular

account number may be listed by issuing a present command. On the 2740 terminal will appear the file names created for that user account number under which the 'present' command has been issued. Associated with each file name will be data specifying the last day on which the lesson file was used.

files:

A 'files' command will produce a summary on the use of files and disk space for the user account number signed on.

The Second Set of Commands:

$grades:

A numeric representation of student performance during the course of a teaching session may be obtained by issuing the $$grade command. The returned answer shall indicate the number of tries at questions, the number of correct replies, and the number of erroneous answers:

TRIES 050 RIGHT 038 WRONG 012

$$end:

A $$end command will break communication between

the user and the lesson file. The lesson file remains saved in the system for future access.

$$abandon:

The entire lesson file is deleted fron the system and the tracks associated with the file are released and made available to any new file being generated.

$$teach n:

This command will put the terminal station in 'teaching' mode and presentation of the lesson will begin. The portion of the lesson file where the presentation is asked to start must be stated in the value 'n'. If the value is not indicated, a value of 1 will be assumed.

$$password:

Identification codes for students who are to have legal access to the lesson file may be generated, along with the file, by using the $$password command. The system will request input of the student passwords, one at a time:

P001 ?

The teacher is asked to enter the password for the first student. The password may assume any character

combination and can have a length of up to six characters. Excess characters will be ignored. After successful input of the first password, the system will request the second ID, third, and so on:

```
P001    ?#10204
P002    ?#12345
P003    ?me
P004    ?you
P005    ?$pass
        etc...
```

A maximum of fifteen student passwords may be included with any one lesson file. When the password buffer is filled, the following message will appear on the terminal:

PASSWORD BUFFER FULL

$$plist n,m:

A $$plist command will list the contents of the password buffer starting at the n th password and continuing for m consecutive password entries.

$$preplace n:

To replace the n-th password in the buffer, a $$preplace n command is required. The system will request

entry of the new password:

P00n ?

$$pmaximum n:

A $$pmaximum command will indicate that only n passwords have been entered in the password table (buffer).

$$map:

The lesson map, defining the sequence of steps followed during the presentation of lesson material, may be generated along with the file by entering a $$map command. The system will respond with the following:

M001 ?

The teacher is asked to enter information pertaining to the first question or portion of lecture material. The information entered assumes the following format:

Line# r1 r2 r3 r4 r5 cr S slide#,cnt T tape#

where,

a. Line# refers to the number of the line of lesson text stored on disk where a question or portion of lecture material begins.
b. ri (i=1,...,5) represents the question or portion of lecture material to which a branch must take place if

the student's answer corresponds to a digit 1 to five, respectively. If a '0' is entered at any Ri, then the system will assume that extra explanation is to be given to the student.

c. cr defines the correct reply to the question.

d. the character 's' stands for slides.

e. slide# defines the first of a consecutive series of slides to be displayed along with the question or portion of lecture material.

f. cnt is the count of slides in the consecutive series.

g. the character 't' stands for tape recorder.

h. tape# defines the tape message to accompany the question or portion of lecture material.

A typical example of proper entry in the lesson map might be:

M008 ?11,7,4,2,3,1,2,s10,3,t35

The above is interpreted as:

"...Question number 8 begins in line 11 of the lesson text stored on disk. Included in question number 8 are the slides (10,11,and 12) to be displayed and the pre-recorded tape message number 35 to be played. The student must answer '2' as the correct reply to the question. If he fails to do so, then output the material of question 7 if he answered '1', output the material of question '2' if he answered '3', and so on.

A maximum of 255 questions or separate portions of lecture material may be structured in this way. When the lesson map buffer becomes filled to capacity, the message to appear on the terminal will be:

QMAP BUFF FULL

Each row of the lesson map is numbered consecutively - first the letter M appears, followed by the question number and a '?' mark to request entry on the part of the user:

```
 TEACH1                    USER
M001      ?  112 1 4 3 5 2 4 s10,1 t22
M002      ?  124,0,3,6,3,1,3
M003      ?  256 4 1 2 3 7 1 s20,3 t19
M004      ?   etc...
```

It is not necessary to input slide or tape recorder data if none exist, as evidenced in row 2 of the lesson map above.

A better insight, on the lesson map, may be obtained by studying figure 8, showing a typical teacher/system session in the generation of information for a new lesson file.

$$mlist n,m:

Contents of the lesson map may be displayed with the help of a $$mlist command. Listing will begin at the

n-th row of the map and continue for m consecutive rows.

```
ENTER ACCT #,PRG NAME
30 teach1                                   ← CALL IN TEACH1 APPLICATION PROGRAM
TEACH SYSTEM IN CONTROL * ENTER REQUEST
?n lesson.file                              ← GENERATE A NEW LESSON FILE
PRECEDE ALL SUBSEQUENT COMMANDS BY $$
?$$map                                      ← DEFINE THE MAP FOR THE LESSON
M001     ?1 2 2 0 2 2 3 s16,2
M002     ?2 6 3 4 4 5 1
M003     ?14 4 6 4 4 4 2
M004     ?16 5 6 5 5 5 2
M005     ?18 6 2 2 2 2 1
M006     ?20 7 7 7 8 7 4 s62,1
M007     ?24 8 1 2 2 2 1
M008     ?$$mm 7                            ← REPLACE MAP LINE # 7.
?$$mr 7
M007     ?24 8 2 2 2 2 1
M008     ?27 7 1 7 7 7 7
M009     ?$$mm 8
?$$mr 8
M008     ?28 7 1 7 7 7 2
M009     ?29 11 13 101 10 10 2
M010     ?31 11 13 10 10 10                 ← NO CR SPECIFIED IN LAST MAP ENTRY
NO CORRECT REPLY SPECIFIED
M010     ?31 11 13 10 10 10 2
M011     ?34 13 12 12 12 12 1
M012     ?37  9 37 37 37 37 1
M013     ?38  1 14 1 1 1 2 s4,1
M014     ?40,1,1,1,1,1,1
M015     ?43 1 1 1 1 1 1
M016     ?$$mm 15                           ← DEFINE MAX # ENTRIES IN MAP
?$$ml 1,15                                  ← LIST CONTENTS OF MAP
M001      SEQ# 0001  R 002  002  002  002  002  CR 03 S 16 # 02  T  0000
M002      SEQ# 0008  R 006  003  004  004  005  CR 01 S 00 # 01  T  0000
M003      SEQ# 0014  R 004  006  004  004  004  CR 02 S 00 # 01  T  0000
M004      SEQ# 0016  R 005  006  005  005  005  CR 02 S 00 # 01  T  0000
M005      SEQ# 0018  R 006  002  002  002  002  CR 01 S 00 # 01  T  0000
M006      SEQ# 0020  R 007  007  007  008  007  CR 04 S 62 # 01  T  0000
M007      SEQ# 0027  R 007  001  007  007  007  CR 07 S 00 # 01  T  0000
M008      SEQ# 0031  R 011  013  010  010  010  CR 00 S 00 # 00  T  0000
M009      SEQ# 0000  R 000  000  000  000  000  CR 00 S 00 # 00  T  0000
M010      SEQ# 0031  R 011  013  010  010  010  CR 02 S 00 # 01  T  0000
M011      SEQ# 0034  R 013  012  012  012  012  CR 01 S 00 # 01  T  0000
M012      SEQ# 0037  R 000  037  037  037  037  CR 01 S 00 # 01  T  0000
M013      SEQ# 0038  R 001  014  001  001  001  CR 02 S 04 # 01  T  0000
M014      SEQ# 0040  R 001  001  001  001  001  CR 01 S 00 # 01  T  0000
M015      SEQ# 0043  R 001  001  001  001  001  CR 01 S 00 # 01  T  0000
?$$mr 9
M009     ?29 11 13 10 10 10 2
M016     ?$$ml 9,1
M009      SEQ# 0029  R 11   13   10   10   10   CR 02 S 00 # 01  T  0000
?$$s s                  ← DEFINE ORDER OF A/V MATERIAL PRESENTATION
DEVICES DEFINED
?$$password             ← GENERATE STUDENT IDENTIFICATION CODES
P001     ?ginobr
P002     ?102049
P003     ?123456
P004     ?me
P005     ?you
P006     ?him
P007     ?her
P008     ?johhny
P009     ?himtoo
P010     ?everyoneelse
P011     ?$passw
P012     ?they
P013     ?178943
P014     ?xyz
P015     ?near
PASSWORD BUFFER FULL    ← PASSWORD BUFFER LIMIT REACHED
?$$n 102049             ← ONCE PASSWORDS ARE GENERATED, ONE MUST
OK. PROCEED               SIGN ON HIS ID CODE IN ORDER TO
?$$pl 1,17                CONTINUE TO HAVE ACCESS TO THE LESSON
P001      102049
P002      102049
P003      123456
P004      ME
P005      YOU
P006      HIM           ← GENERATE CONTENTS OF STUDENT ID
P007      HER             PASSWORD BUFFER
P008      JOHHNY
P009      HIMTOO
P010      EVERYO
P011      $PASSW
P012      THEY
```

Figure 8

'Create' Mode

```
P013      178943
P014      XY7
P015      NEAR
?$$h lock                ← BAR ACCESS TO PRIVILEGED INSTRUCTIONS
PRIV INSTR NOW INACCESSIBLE
?$$ah                    ← A PRIV. INSTRUCTION WILL NOT BE EXECUTED
THAT IS A NO-NO!
?$$end                   ← CLOSE AND SAVE FILE
13M 00S
ENTER ACCT #,PRG NAME
30 teach1
TEACH SYSTEM IN CONTROL * ENTER REQUEST   ← LIST STATUS OF FILES
?files
MAX # OF FILES= 12 , 02 USED; MAX # OF TRACKS= 80 , 03 USED
?present                 ← LIST LESSON FILES AND LAST DATE USED
LESSON 72097 ,LESS1  72097 ,
NO MORE FILE NAMES
?k lesson.file  key1     ← CHANGE THE KEY ON THE LESSON FILE
KEY UPDATED
?o lesson.key1           ← ACCESS THE LESSON FILE
000001$$pl 1,4
THAT IS A NO-NO!
?$$u lock                ← OPEN WAY TO PRIVILEGED INSTRUCTIONS
PRIV INSTR NOW ACCESSIBLE
?$$n they
OK. PROCEED
?$$pl 1,4
P001      THEY49
P002      102040
P003      123456
P004      ME
?$$end
2M 21S

ENTER ACCT #,PRG NAME
30 teach1
TEACH SYSTEM IN CONTROL * ENTER REQUEST
?o lesson.key1           ← ACCESS OLD LESSON FILE
000043$$t 1              ← START THE TEACHING PROCEDURE ON ITS WAY
?follow-up               ← A SLIDE IS PRESENTED / CONTINUE WITH THE LESSON
?follow-up
THIS IS SAMPLE MATHEMATICAL QUIZ FOR ELEMENTARY GRADES.
What is the sum of 6  +  15 ?
1. 25
2. 13
3. 23
4. 10
5. None of the above.     ← LESSON TEXT STORED ON DISK
ANS.
?3   ← STUDENT ANSWER
OK
The sign '+' defines the process of multiplication.
1. True    2. False
ANS.
?1
WR
Then what does 'x' signify ?
1. Multiplication   2. Division
3. Addition         4. Subtraction
ANS.
?e       ← END TEACHING SESSION
?$$t 1   ← START TEACHING SESSION AGAIN ON ITS WAY
?f
?f
THIS IS SAMPLE MATHEMATICAL QUIZ FOR ELEMENTARY GRADES.
What is the sum of 6  +  15 ?
1. 25
2. 13
3. 23
4. 10
5. None of the above.
ANS.
?2
WR
Your reply was incorrect.
Let's try the problem once more, step by step
What is 5  +  5  ?
1. 10   2. 9
3.  4   4. 0
5.  0
ANS.
?2
WR
5 + 3 add up to 8. Now what is 5 + 5 ?
1. 9   2. 10
```

Figure 8

'Teaching' Mode

And So On.

$$mreplace n:

The contents of the n-th row may be altered, if the lecturer wishes changes to that row. A $$replace n command will achieve this purpose. The system will request the new entries for that row by:

M00n ?

$$mmaximum n:

This instruction, if used, will inform the system that only n separate questions or portions of lecture material are to be considered generated during presentation in a teaching session.

$$name password:

If a password list is inserted in the lesson file to restrict access of the file to certain students, the latter, in turn, must enter their assigned password in order to obtain the lesson. This they do by issuing a $$name command followed by their password. Assuming a proper entry, acknowledgement of the availability of the file will be revealed in the message:

OK. PROCEED

$$setup s,t or $$setup t,s

Whether slides should precede the playing of tape messages or vice versa, is indicated by 's' preceding 't' (or, respectively, 't' preceding 's') in the $$setup command. The order may be changed at anytime. When it comes to actual presentation of the lesson, any tutorial information stored on the projector or tape recorder will be ignored if the $$setup command is never issued.

$$bar lock

Certain instructions that manipulate the structural form of the lesson file are considered to be privileged and, therefore, can be protected against outside tampering by placing a 'lock' on them. The 'lock' is a four letter password that may assume any character combination. The $$bar instruction will render inaccessible the following instructions:

$$setup

$$password

$$plist

$$preplace

$$map

$$mlist

$$mreplace

$$mmaximum

```
$$abandon
$$bar
```

$$unlock lock:

A $$unlock lock instruction will retrieve the full privileged use of the commands listed above. The character combination making up 'lock' must only be known to the author of the lesson since he is the only one who may wish to effectuate changes to the lesson file and to protect his file.

The Third Set of Commands:

follow-up:

When a student has finished examining a slide carefully or listening to a pre-recorded tape message, he may resume normal continuation of the lesson by entering a follow-up command.

repeat:

Questions and other tutorial information may be repeated upon issuing a repeat command.

end:

The end command will terminate the 'teaching' session and will place the terminal station back in 'create' mode.

CHAPTER V

TEACH1 PROGRAM DESIGN

Structure of the TEACH1 application program is modular in nature. Numerous service modules, concatenated to program-dependent data areas and buffers, perform those managing functions which process user requests entered at the teaching station. The modules are divided into two groups. The first of the groups consists of those modules which become active when the teaching station enters 'create' mode:

a. MUM REQUEST PRE-SERVICER
b. COMMAND INTERPRETER
c. MODULE ACCESSOR
d. FILE REQUEST SERVICER
e. LESSON MAP MANAGER
f. PASSWORD PROCESSOR
g. TEXT PRESENTATION MANAGER
h. SLIDE PROJECTOR MANAGER
i. TAPE RECORDER MANAGER

As well as being members of the first group, modules g,h, and i, are active when the terminal station is in 'teaching' mode. They constitute the members of the

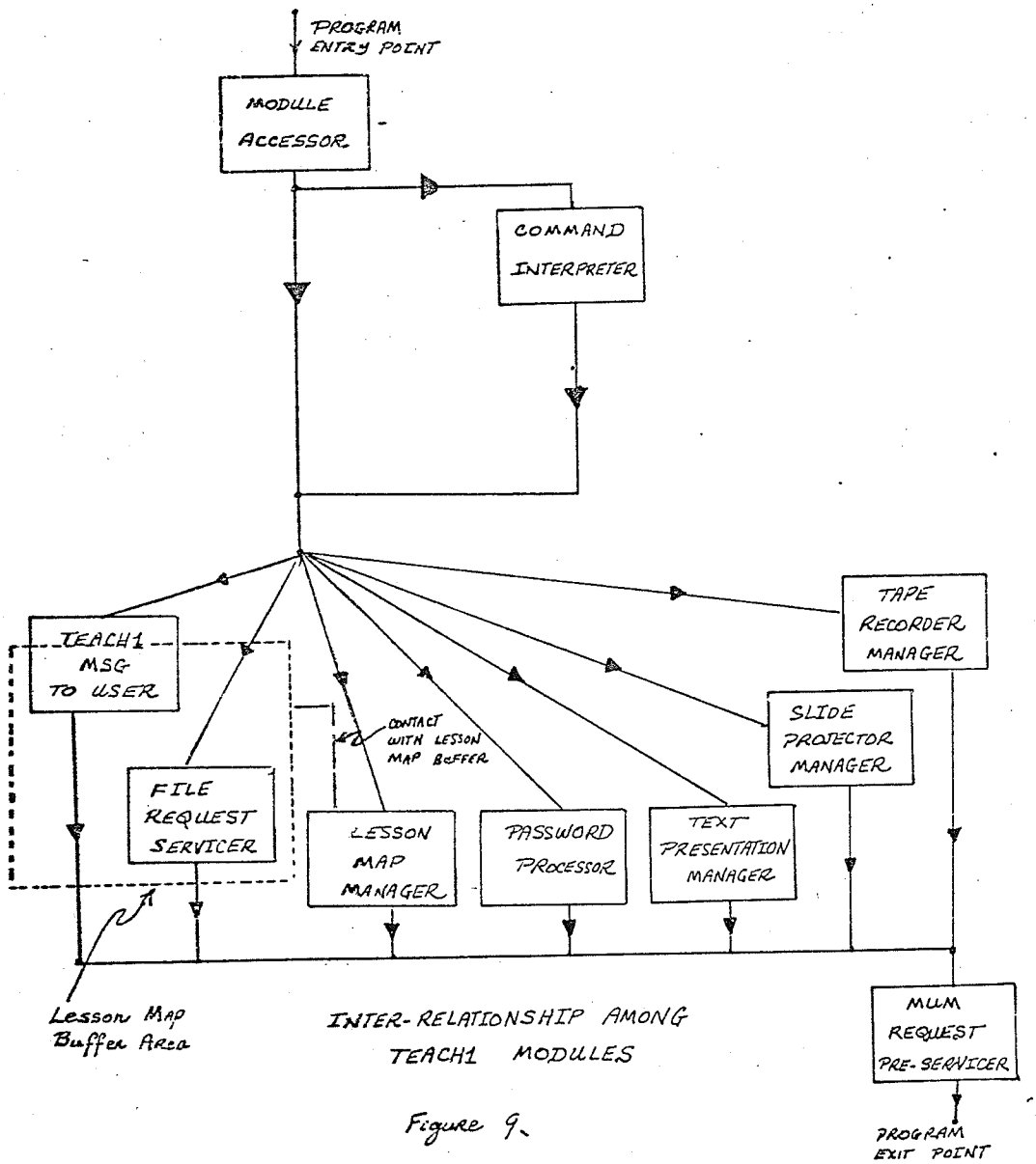
second group.

In order to cut down on the number of executable statements and, in consequence, decrease the response time, data areas were used whenever possible since these only matter at assembly time and not during execution. Most data areas are tables of flags, address constants, and pre-formatted messages to the terminal user, as well as buffers for I/O, lesson map, and student passwords.

As explained earlier, the size of any application program in this system is limited by the dimension of the roll-in/roll-out area in core in which the program is placed when in execution; namely, 7294 bytes.

Just as in most of the other application programs of the TEACH system, TEACH1 overlays all lesson map and password buffers on top of the File Request Servicer coding, once the lesson file is generated.

The inter-relationships of the modules is shown in figure 9. A flowchart of program logic appears in figure 10.



INTER-RELATIONSHIP AMONG TEACH1 MODULES

Figure 9.

BLOCK FLOWCHART
FOR
TEACH1 PROGRAM LOGIC

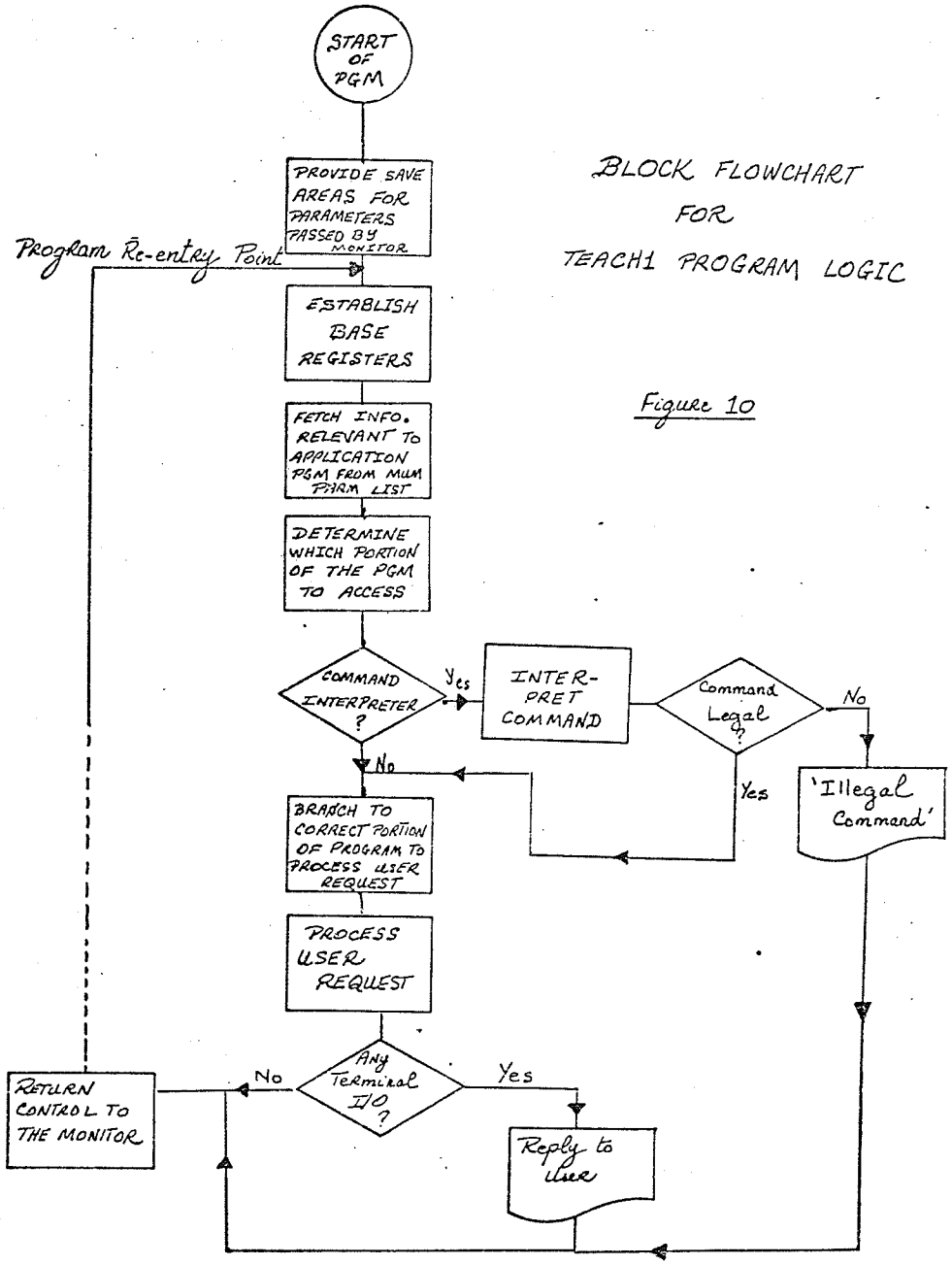


Figure 10

A. The MUM Request Pre-Servicer

Most of the modules issue, at one time or another, some sort of request to the TEACH system monitor. These are mainly requests for I/O to the terminal and special demands to the lesson file manager of the system monitor. Each request has associated with it a code identifying the type of request made. On entering the MUM Request Pre-Servicer routine, this code is already defined. In a save area set aside to contain parameters common to both the application program in execution and the monitor routines, the MUM Request Pre-Servicer stores the following:

a. the address of the output buffer containing the message intended for the receiving terminal station.
b. the address of the input buffer to contain any reply.
c. the lengths of the respective input/output buffers.
d. the address of the lesson file handling parameters.

From the same save area is fetched the entry point address to the monitoring programs which service requests from application programs. Prior to a branch being taken to this address, a return-address is pre-defined for re-entry to the TEACH1 program.

B. Command Interpreter

Any input received from the terminal station is scanned to determine whether the input is user data (e.g., lesson map digits, passwords) or user-issued commands (e.g., create map, start teaching procedure). In any case, if the input is comprehensible to the Command Interpreter, a branch takes place to the proper servicing modules. An illegal or incomprehensible input will result in a request to the user to re-submit his message.

When the monitor passes control to the TEACH1 program, it also makes available to the latter a value for the length of the input message. The length message is used to delimit how far scanning of the input message is to proceed. This has two effects on the nature of the program: (1) the input buffer does not have to be scanned from one end to the other for an input message of few characters; and, (2) the input buffer need not be filled with blanks prior to receiving the input message each time. The time saved is considerable.

C. Module Accessor

All modules of TEACH1 turn on program flags before calling on the MUM Request Pre-Servicer. These flags are used by the Module Accessor to determine which routines are

to become active when a return from the request servicing monitor programs is made. A one-to-one relationship exists between the flags and a table containing the relative addresses of the TEACH1 routines. The address associated with the last flag set before control was passed to the monitor is fetched by the Module Accessor to become the virtual re-entry point when control is returned to the TEACH1 program.

The Module Accessor searches a table of twenty flags to find the one entry in the table which matches with the last flag set.

D. The File Request Servicer

The File Request Servicer processes requests for the generation of a new lesson file, the insertion of a protection key to the file, accessing an old lesson file, displaying file status for files created, and changing the protection key of a file.

In the case of the generation of a new lesson file, a parameter list is prepared for eventual submission to the Monitor which actually puts aside the disk tracks to contain the file. This parameter list consists of a code for file creation, the user account number for which the file is being created, the time of file creation, status of card images in the file, the name by which the file is to be

called, and the protection key for the file (if any).

Accessing an old lesson file or changing a key on the file demands that information similar to that contained in the parameter list above be submitted to the Monitor. The specifications not required in this case are the time of creation of the file and the status of the card images in the file.

Displaying lesson files status requires that a special code be sent to the Monitor. The Monitor, then, forces into the terminal buffer information pertinent to the files under a specific account number. The pertinent information may be file names, last date each file was used, and number of tracks occupied by the files. Control is then returned to the Module Accessor which, in turn, calls the File Request Servicer to scan the data transferred to the terminal buffer and provide suitable descriptive output to the user.

E. The Lesson Map Manager

The Lesson Map Manager converts the space occupied by the File Request Servicer into a buffer to contain the map matrix for the lesson. The File Request Servicer code is no longer required once a file has been generated and that the application program becomes part of the file. Therefore, in its place can be maintained table entries affiliated with

the sequence of steps to be followed in the dessemination of lesson material.

Each row of the map matrix contains data relevant to a question or other text content of the lesson. The first entry in the row depicts the line number for the beginning of the information entity (e.g., a question). Immediately following are five entries defining remedial action to be taken for each of the five possible replies by students (1,2,3,4,or 5). In addition, the correct answer to the question, the number of slides to be displayed, the number of the first slide in each sequence of slides, and the number of the tape message to accompany the presentation are included in each row. All in all, ten entries comprise each row of the map matrix.

The Lesson Map Manager scans the map data entered at the terminal, converts the data to binary representation, then concatenates the binary information to produce a row of the map matrix. All ten entries of the row occupy a total of eleven bytes.

The Lesson Map Manager also provides the facilities by which sections of the map could be replaced or listed as well as defining the maximum number of rows in the matrix.

F. The Password Processor

It is often found necessary to restrict use of

lesson files to students registered in the course for which the lessons have been prepared. For this reason, a password processor built into the TEACH1 application program, creates a buffer to contain a list of student identification codes entered by the teacher at the time of lesson file generation.

When a student signs on to the system, he is requested to enter his ID upon which time the Password Processor takes over and attempts a match with the list of student identification codes. A successful match opens a key to the lesson file.

In order to protect against illegal tampering of lesson file content, the Password Processor guards against prohibited access to privileged intructions, reserved only for the use of the author of the lesson file.

The Password Processor makes available editing facilities for alterations to the list of identification codes.

G. The Text Presentation Manager

The Text Presentation Manager acts as a buffer between the user and the lesson material stored on disk. It calls on the file-handling program of the Monitor to fetch the lesson material from disk. The material is then scanned to determine whether or not to treat it as extra

explanation. In either case, it is eventually sent to the terminal station where it is printed on the 2740 teletype.

The Text Presentation Manager consists mainly of a succession of Read/Write procedures.

H. The Slide Projector Manager

The Slide Projector Manager formats messages to be sent to the 2968 control unit for the display of slides. These messages are five bytes long. The first two bytes, consisting of an unprintable control character prefix followed by a lower case letter 'o', denote to the receiving terminal station that the message is intended for the 2968 control unit and not the 2740 telecommunications terminal. The third byte defines the audio/visual device under consideration; namely, the slide projector. The last two bytes contain a lower case character combination identifying the specific slide to be displayed.

All characters included in the message are given in EBCDIC internal representation. Conversion to 2740/2968 terminal combination BCD code was not attempted in the application program since such translation is undertaken by the Monitor.

Status replies from the 2968 control unit are evaluated to determine whether the projector is already executing a previous instruction, if the projector is

off-line, or if the projector is ready to accept the next intruction. In the case where the projector is busy executing a previous instruction, the Slide Projector Manager goes into a loop sequence until a signal received from the 2968 control unit conveys readiness to accept the next instruction. If the projector is off-line, the user is requested to turn the projector on. Request for status from the 2968 control unit is achieved when the slide projector manager demands display of slide # 0.

When this module has terminated its assignment, it passes on control to the Tape Recorder Manager or Text Presentation Manager, whichever the case may be.

I. The Tape Recorder Manager

The Tape Recorder Manager performs functions similar to those of the Slide Projector Manager in that it formats messages to be sent to the 2968 control unit, for the playing of taped information. The messages sent to the tape recorder are almost identical in form to the ones meant for the slide projector. The differences lie in:

(1) the device code stored in the third byte of the message, defining the audio/visual unit.

(2) the address combination depicting the tape message to be played.

The lower case character combination for the

address of the tape message is a relative address, relative to the address of the last tape message played. This is not the case for the slides, where the address specified is absolute. The address of any slide is independent of the last slide shown.

Status replies from the 2968 control unit indicate if the tape recorder is ready to accept an instruction, busy executing another instruction, off-line, or if the tape reel is at the overrun position. The user is informed of a condition existing with the tape recorder if this condition is external to the control of the Tape Recorder Manager. Waiting for the tape recorder to become available, the Tape Recorder Manager goes into a loop sequence, as well.

APPENDIX A

OTHER TEACH1 MESSAGES

1. TEACH SYSTEM IN CONTROL*ENTER REQUEST

The user may issue any command from the first set of commands.

2. UNIT NON*EXISTENT

The audio/visual unit(s) defined in the $$setup command does(do) not exist.

3. NO MORE FILE NAMES

No more lesson files currently exist under this user account number.

4. ILLEG. COMMAND

Self explanatory.

5. NO FILES EXIST CURRENTLY

No lesson files currently exist under this user account number.

6. PRIV INSTR NOW INACCESSIBLE

A 'lock' has been placed on all TEACH1 privileged instructions.

7. PRIV INSTR NOW ACCESSIBLE

The 'lock' has been removed from the privileged instructions and any one may have access to them.

8. DEVICES DEFINED

The order of audio/visual material presentation has been

successfully defined.

9. PROJ OFF-LINE

The power of the slide projector has not been turned on.

10. REPLACE SLIDE TRAY

Self explanatory.

11. TAPE REC OFF-LINE

The power of the Uher tape recorder has not been turned on.

12. REPLACE/REWIND TAPE REEL

Self explanatory.

13. PRECEDE ALL SUBSEQUENT COMMANDS BY $$

The terminal station has now entered 'create' mode and all instructions issued must belong to the second set of commands.

14. END OF LESSON

The end of the lesson has been reached. The user may input any instruction from the second set of commands.

15. ANS. ?

The user is asked to provide a reply to the question put forward.

16. CR

Correct reply.

17. IR

Incorrect reply.

18. NO SUCH ANSWER

The reply given to a question is not valid.

19. KEY UPDATED

A request to change the key on a file name has been processed.

20. ILLEG USER ID

No such student password exists; and, hence, access to a lesson file will not be possible.

INSERTION OF LESSON MATERIAL ON A DISK LESSON FILE (Generated Under TEACH1)

On-line insertion of tutorial information, when the TEACH1 application program assumes control of the system, is impossible. The following procedure should be observed in storing tutorial information on a lesson file residing on disk and generated under control of the TEACH1 application program:

1. Create and prepare a lesson file while the TEACH or any other text editing application progam is on-line, inserting all tutorial information to be included in the lesson.
2. Obtain a 'hard' (punched deck) copy of the lesson file by using the EDITLIST utility program (see Appendix B).
3. Delete the file.
4. Request on-line execution of the TEACH1 application program, on the TEACH system.
5. Generate a new lesson file and disk space to contain

the lesson file by issuing a 'new filename.key' command.

6. Sign off the system.
7. Load the punched deck, containing the lesson material, onto the newly generated file, by using the FILETOED utility program (see Appendix B). Then, the material becomes available for its on-line presentation under control of the TEACH1 application program.

Any type-setting commands, included in the lesson material, are ignored by TEACH1, with the exception of the # sign which defines extra explanation to a question or other lecture content.

APPENDIX B

JCL REQUIRED FOR TEACH SYSTEM MAINTAINANCE

1. To load the system into core and execute:

```
//TSYSTM  JOB  '............'
/*SETUP  THIS JOB REQUIRES 40K OF CORE, NO CPU TIME
//MUM  EXEC  PGM=MASTPROD,REGION=40K
//SYSUDUMP DD SYSOUT=A
//STEPLIB DD DSN=MUM.CS.LOADMODS,DISP=SHR
//FNREC   DD DSN=MUM.CS.ACCT,DISP=SHR
//CARDR   DD DSN=MUM.CS.ROLLIB,DISP=SHR
//ROLLIB  DD DSN=MUM.CS.ROLLIB,DISP=SHR
//TPDD051 DD UNIT=081
/*
```

2. To load a lesson file onto the system from cards:

```
//LOADFILE  JOB '............'
//  EXEC  PGM=FILETOED,REGION=30K
//CARDR DD DSN=MUM.CS.EDITCARD,DISP=SHR
//FNREC DD DSN=MUM.CS.ACCT,DISP=SHR
//LIBDD DD DSN=MUM.CS.ROLLIB,DISP=SHR
//STEPLIB  DD  DSN=MUM.CS.PROGS1,DISP=SHR
//SYSPRINT DD SYSOUT=A
//FNAMES DD *
```

```
User account number.password filename.key
/*
//DSFTE1 DD DATA
                |
                |
                |
                .
            card file
                1
                |
/*
```

3. To load an application program into the roll library:

```
//LOAD JOB '..........'
//  EXEC   MUMCLD,NAME="name(R)',ID=CS
//ASM.SYSIN DD *
                |
                |
                |
                .
          application program
                |
                |
                |
                |
                |
/*
```

4. To obtain a hard copy of a lesson file:

```
//PUNCHFIL JOB '............'
//   EXEC  EDITLIST,ID=CS
//GO.FNAMES DD *
User account number.password  filename.key
```

```
/*
//GO.SYSPUN1 DD SYSOUT=B
```

SOURCES CONSULTED

[1] "Computer-Aided Learning". Science Dimension, April, 1970.

[2] Brahan, J.W. "Computer-Aided Learning". Bulletin of Radio and Electrical Engineering Division, National Research Council, Vol. 19, No. 3, pp. 12 ff, 1969.

[3] Brahan, J.W. The Computer-Aided Teaching Study at the National Research Council, Paper presented at the Canadian Psychological Association Convention, University of Calgary, June, 1968.

[4] Grieve, T.D. Use of the National Research Council's Computer-Aided Teaching System, Paper prepared by Information Science Section, Radio and Electrical Engineering Division, National Research Council, Ottawa, August, 1968.

[5] Hlady, A.M. "An X-Y Touch Sensitive Position Encoder for Computer Input". Bulletin of Radio and Electrical Engineering Division, National Research Council, Vol. 19, No. 3, pp. 15 ff, 1969.

[6] Symonds, M. Computer Detection of Misspelled Words, pub. Radio and Electrical Engineering Division, National Research Council, ERB-843, Ottawa, August, 1970.

[7] Brief History of Computer-Aided Instruction, Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, 1969.

[8] Progress Report Stanford Program in Computer-Assisted Instruction, Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, March, 1970.

[9] IBM Coursewriter III for System/360, Version 2, IBM Applications Description Manual, H20-0587-1, 3rd ed., August, 1969.

[10] Morrison, H.W.,and Adams, E.N. "Pilot Study of a CAI Laboratory in German". The Modern Language Journal, Vol. LII, No. 5, May, 1968.

[11] Morrison, H.W., Adams, E.N. and Reddy, J.M. "Conversation with a Computer as a Technique of Language Instruction". The Modern Language Journal; Vol. LII, No. 1, January, 1968.

[12] Molnor, A.R. and Sherman, B. U.S. Office of Education Support of Computer Activities, pub. U.S. Government Printing Office, Washington, 1969.

[13] "There's a Computer in Your Future", reprinted from American Education, pub. by U.S. Department of Health, Education, and Welfare,/ Office of Education, November, 1967.

[14] Brahan, J. W. Automation and Education/A Review, pub. Radio and Electrical Engineering Division, National Research Council, ERB-765, Ottawa, June, 1967.

[15] Bushnell, D. D. and Dwight, W. A. The Computer in American Education, pub. John Wiley and Sons, Inc., New York, 1967.

[16] Computers and Education, ed. R. W. Gerard, pub. McGraw Hill Book Company, 1967.

[17] Programmed Teaching: A Symposium on Automation in Education, ed. Joseph S. Roucek, pub. Philosophical Library, Inc., New York, 1965.

[18] Reid, B. The Design and Implementation of a Remote Terminal Monitor, pub. Department of Computer Science, University of Manitoba, Winnipeg, February, 1972.

[19] Abraham, C. Manitoba University Monitor, University of Manitoba Publication, Scientific Report No. 1, Winnipeg, September, 1970.

[20] IBM 2968 Model 11 Audio/Visual Control: Component Description and Operating Procedures, IBM Manual GA26-1593-1, 2nd ed., August, 1970.

[21] Rugger, K. MUM Implementation Guide, pub. Department of Computer Science, University of Manitoba, Winnipeg, February, 1971.

[22] Abraham, C. Real-Time TABLE Program, University of Manitoba Publication, Scientific Report No. 17, Winnipeg, 1971.

[23] Abraham, C. Real-Time INDEX Program, University of Manitoba Publication, Scientific Report No. 19, Winnipeg, 1971.