Behavior Research Methods & Instrumentation
1975, Vol. 7 (2), 107-112

SESSION V
CONTRIBUTED PAPERS: SYSTEMS AND TECHNIQUES

DONALD I. TEPAS, *Saint Louis University, Presider*

The CLIPR display terminal experiment system

TERRY L. SPEAR, DENNIS OVERGARD, and THOMAS W. CHRISTIAN
Computer Laboratory for Instruction in Psychological Research
University of Colorado, Boulder, Colorado 80302

The Display Terminal Experiment System (DTES) is a specialized system for preparing, executing, and analyzing experiments involving the presentation of text materials and the recording of response latencies. The system includes: materials preparation routines, an experiment description language and translator, a real-time experiment monitor program, and data recovery and data analysis programs. DTES is intended to enable users with little or no computer programming experience to construct and run sophisticated real-time experiments. The major features of this system are: (1) a complete package for handling every phase of the experiment from materials preparation to elementary data analysis, and (2) the experimenter is permitted to run an arbitrary number of subjects concurrently with each in a different experimental condition.

The CLIPR Display Terminal Experiment System (DTES) is a specialized system designed to simplify the automation of a general class of psychological experiments involving presentation of verbal materials, recording discrete multiple choice responses, and measuring of response latencies.

DTES was developed in response to the need of a group of psychologists to utilize the powerful experiment control and data collections facilities available at CLIPR in experiments requiring the handling of large amounts of text. Previously, technical knowledge of computers in general, and the CLIPR system in particular, was necessary for a psychologist to successfully implement a particular paradigm in a reasonable length of time. The amount of technical information with which a user had to become familiar was often enough to discourage the noncomputer oriented experimenter from using the powerful facilities available even for those paradigms ideally suited for computer implementation. This problem is well known, and several programming systems that attempt to ease the computer system-psychologist interface have been described (McLean, 1969; Polson & Campbell, 1972). The design of DTES, while not attempting a general solution, deals with this problem and others faced by the experimenter within the class of experiments previously described.

DTES was designed with the following goals in mind: (1) little technical knowledge needed to use the system; (2) "complete system" approach; (3) total automation of experiment; (4) guaranteed accuracy of data, from collection through final analysis; and (5) system modularity.

It was felt that a viable system aimed at the noncomputer oriented psychological experimenter must be a complete package. That is, there must be a clearly defined sequence of tasks for a user to follow when implementing an experiment in conjunction with the software packages used to implement each task. Thus, DTES provides software support for every phase of an experiment from materials preparation through elementary statistical analysis of the experimental data.

Automation of a psychological experiment was considered an important goal, since many experiments of the class of experiments capable of being implemented in DTES require little or no experimenter-subject interaction. An experimenter can, with computer automation, run more than a single subject simultaneously. Thus, it is possible for an experimenter to supervise as many as six subjects concurrently, which greatly increases the experimenter's productivity over the typical one-on-one experimental situation. DTES currently provides up to six

Requests for reprints should be sent to Terry L. Spear, Muenzinger E318-CLIPR, Department of Psychology, University of Colorado, Boulder, Colorado 80302.

experimental stations that can be shared among experimenters in any desired configuration.

Accuracy of the collected data must be guaranteed and maintained throughout its existence in the system for the obvious reasons. The measurement of response latencies is made by well tested routines resident in the CLIPR system. These routines have been shown to be accurate to ± 1 msec with high probability. Each data record generated by a subject is stamped with the current date/time and a binary checksum. The programs that subsequently process these data use this information to maintain the security of the data throughout its existence in the system.

Modularity was used in the system design primarily to ease implementation and modification. Each module performs a specific function and can only affect another module by a well defined and restricted data path. The modules include a program to generate materials, an experiment description language translator, a real-time experiment monitor, data recovery, and data analysis programs.

In order to make the system easy to learn and use by the nonprogrammer, each module uses only a basic set of control commands. For example, the experimenter needs to learn only five commands to operate the Display Edit program. Describing an experiment to the system requires learning the 14 statements of the Experiment Description Language. Thus, a naive user realistically can be expected to learn the necessary operations with which to bring up an experiment in a very short time.

The development of an experiment to be run by the system begins with the selection of the material to be presented to the subjects. This material is transferred by the experimenter to a random-access disk file, called the Display File, using the Display Edit program. Information is stored on the Display File in units called records, each of which contains a complete screen of information for a CRT terminal. The records are numbered sequentially from 1 to N, where N is the number of records allocated to the display file. Information on the Display File is always referenced by record number.

Next, the experimenter prepares a description of the experiment to be performed, using statements from the Experiment Description Language (EDL). The EDL statements are generally of a fixed format in which the user supplies parameters. For example, the following EDL statement would cause record 26 of the Display File to be presented to a subject for 12.65 sec.

SHOW 26 FOR 12.65

Also included with the experiment description are: the condition or experiment identifier, the name of the experimenter, the name of the Display File, the name of the file to receive the data generated by the experiment, and, optionally, one or more statements describing the data records to be included in the preliminary statistical analysis that takes place as the data is being transferred to the experimenter's Data File. When the experiment description is complete, it is translated to a machine-readable form by the EDL Translator. After testing and debugging the description, the experiment is ready to run. To run a subject, the experimenter first locates a DTES experiment station not in use. The experimenter, using the display keyboard, enters the name of his Driver File, a four-character subject identifier, and optionally a seed for the random number generator. The experiment contained on the Driver File then begins execution.

FUNCTIONAL MODULES OF DTES

The entire system includes six major autonomous programs that are both logically and physically separate. The modules, as implied, correspond to the automation of various steps required in the development of an experiment. The foreground (real-time)/background (batch) organization of the CLIPR Xerox Sigma 3 (see Bailey & Polson, 1975) was utilized to permit maximum flexible use of the system. Of the six major programs, four are available in the batch environment, one runs as an on-line program (low priority real-time) and one in the real-time environment. The overall system configuration is shown in Figure 1.

The first step of development of an experiment is the materials preparation in a computer usable form. This function is handled by the Display Edit program. Each display is created on the CRT exactly as it will appear to the subject. The display terminal used to create and edit display records is identical to the CRT terminals used in the DTES experiment stations. These display terminals have sophisticated screen editing features built into them and are designed to handle large text displays efficiently. This permits the user to manipulate display records directly, using computer interaction only for storing and retrieving display records from on-line disk files. The Display Editor operates in an on-line mode to allow a minimum allocation of resources, since materials preparation tends to be very time consuming.

Next, the experimenter must describe the experimental condition to the system, describing which displays to present and under what circumstances. A simple language was developed with which the user can straightforwardly describe the experimental condition. The 14 statements of this Experiment Description Language, along with a brief description of their functions, are presented in Appendix 1. The statements were designed to be as simple as possible to allow translation to be performed by an existing language independent macro translator (Waite, 1967) in conjunction with the local assembler. Together, these produce a binary form of the EDL statements that can be executed interpretively. The binary translation of a user's experiment description is saved on the user's Driver File.

The third module of the system is the real-time

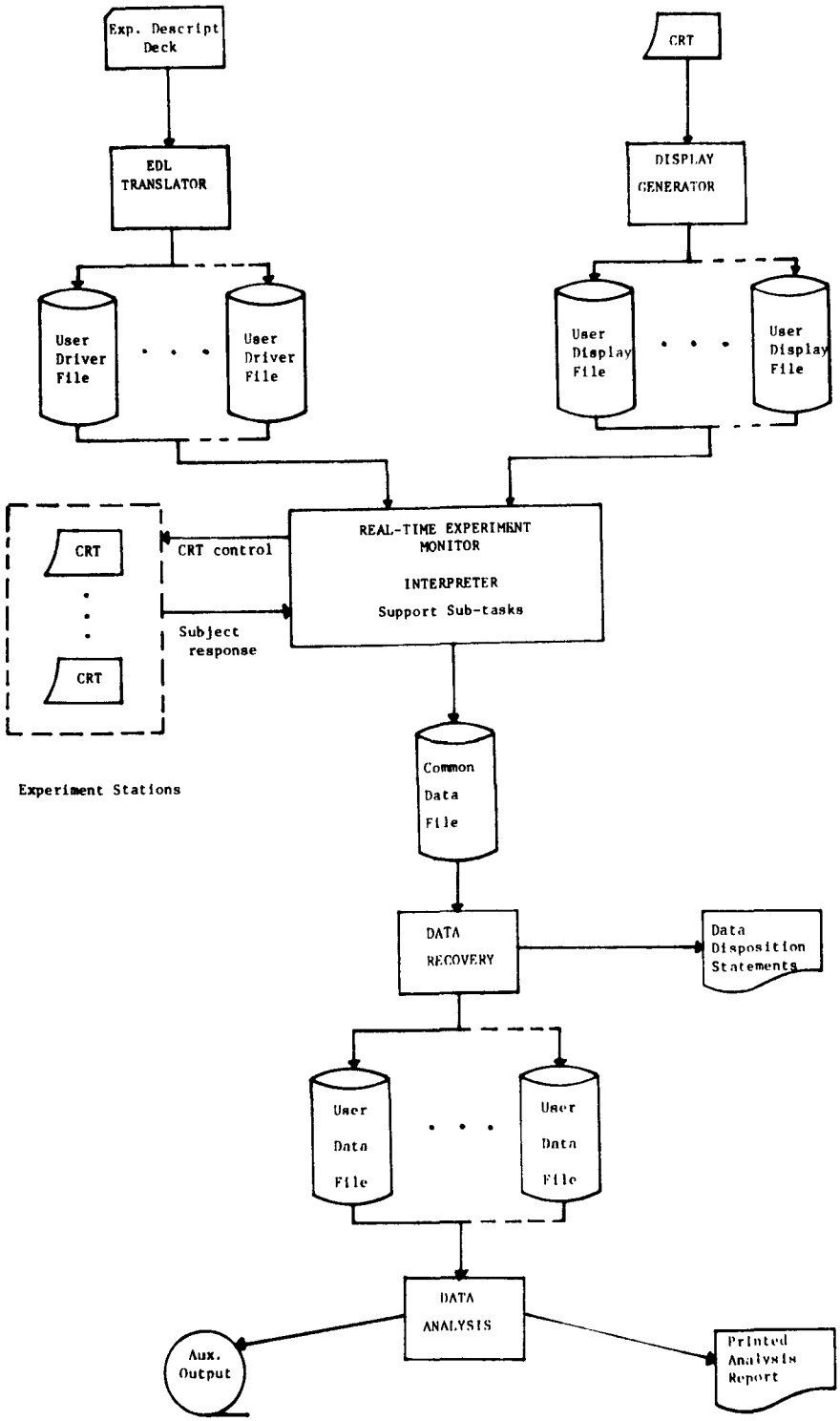

Figure 1. DTES System Configuration.

program that controls the experimental stations and interprets the experiment description. This program is composed of a main task and several support subtasks. The main task contains the experiment monitor and a single copy of the EDL Interpreter. The subtasks perform all of the necessary utility functions such as response collection, timed intervals of delay, and I/O. The interpreter makes requests for these utility functions through the experiment monitor. The interpreter is a single large FORTRAN subroutine written in a reentrant form. Each experiment station on-line with DTES corresponds to a single unique activation of the reentrant interpreter. Thus, each terminal has an associated core overhead consisting only of the local storage required by the interpreter. Currently the system requires approximately 10K 16-bit

words of core to operate a single experiment station. Each additional station requires an additional 270 words of core. Thus, six stations can easily be run in 12K of memory on the Sigma 3 (the current configuration).

An important concept of the real-time experiment monitor/control program is that it executes interpretively the statements comprising an experiment description. Each EDL statement corresponds to an instruction that is executed as a single function of the interpreter. The instructions that define an experimental procedure are kept on a random access disk file, the Driver File. Only a small amount of instructions, usually those comprising the current block being executed, are actually in core at once. This is possible since most EDL statements involve waiting on external events for several seconds. An experiment description does not affect the amount of core storage needed by a particular experiment station running a lengthy experiment.

The interpreter is shared by all experiment stations under control of the experiment monitor. Because it is shared, only a single copy of the interpreter is required in core. The monitor treats each experimental station as a distinct process. A priority queueing structure is used to control the multiplexing of the interpreter efficiently. Before each process (experimental station) begins execution in the interpreter, the monitor establishes the correct environment for that experimental station (the 270-word local store of the interpreter).

Data collected by an experiment are stored initially on a common on-line disk file. Each subject at each station produces one subfile in this common file. At the end of a series of experimental sessions, the real-time experiment monitor is terminated and the common file is separated into the subfiles generated by each subject. This function is handled by the Data Recovery program. The Data Recovery program is run in the batch mode to recover the subject-generated subfiles and to append each to an experimenter's Data File as specified in the experiment descriptions. At the time the subfile is processed by the program, a Data Disposition Statement is generated for each subject (subfile). This statement contains the experimenter's name, the condition identifier, the subject identifier, the current date, the number of data records that were saved, and the seed for the random number generator. The preliminary statistical analysis, if requested, also appears on the Data Disposition Statement. It consists of means, variances, and standard deviations for the data cells selected by the experimenter (see EDL 'PATTERN' statement in Appendix I) and is broken down by response type into several categories (all, correct, incorrect, other) for the experimenter's convenience. At the end of the experimental session, the experimenter has a record and a brief analysis of each of his newest subjects' performances.

The data records on the experimenter's data file can be processed further by the final module of the system, the Data Analysis program. Available in the batch mode, this program enables the experimenter to select any subset of the data contained on the Data File, according to identification information supplied by the experimenter that is matched against identification fields in each data record. The selected data records may then be processed for more complete statistical analysis and the generation of frequency plots. The statistical analysis is similar to that provided by the Data Disposition Statement but adds skew, kurtosis, and the sum of several powers of the variables. It also provides a data breakdown by response numbers. The user can establish lower and upper bounds on the data to be processed, and specify whether information out of bounds is to be ignored or replaced by a specified value. The Data Analysis program also offers two machine readable data output formats, which can be used to transport the data for further analysis and for management of the experimenter's Data File. Thus, since the experimenter need not actually handle the data (i.e., keypunching the data for computer analysis), a major source of possible error has been eliminated.

STANDARD DTES EXPERIMENT STATION

The current DTES experiment station consists of an alphanumeric CRT display terminal and keyboard, an 8-bit digital input line and special hardware CRT control electronics. The CRT terminals have display screen sizes of either 12 lines of 80 characters or 16 x 64. The CRTs have been specially modified to permit the scan beam to be turned off (blanked) and on (unblanked) by computer control. This is an important feature, since many CRTs are being driven from the same I/O channel and there are usually unpredictable delays associated between requesting an output transfer and the actual appearance of the text on the screen. In a shared central processor environment, even more delay may occur before the program which requested the output knows that the transfer is complete (Christian & Polson, 1975). When the CRT scan beam is turned on by the computer, the special control hardware sends a signal back to the computer when the scan is at a fixed location. Thus, it is possible for the controlling program to know exactly when the stimulus material has appeared on the screen. Associated with the blanking unit is a stimulus onset warning device. This device is selected by the experimenter to provide: (1) no stimulus onset warning; (2) an audible warning; (3) a visual warning; or (4) both visual and audible warnings. The fixed period between the warning and the stimulus onset (CRT unblanking) can be preset to the desired interval. This same device can also optionally control the duration of stimulus presentation (i.e., blanking the CRT some interval after it has been unblanked). The timing circuitry used for the CRT control is totally independent of the computer's timing for dependable accuracy. Thus, the sophisticated CRT control device can, in the most complicated configuration, perform the sequence of: (1) present warning signal, (2) delay for some interval, (3) unblank the CRT, (4) delay for a second interval, and (5) blank the CRT.

Table 1
An Example of Experiment Development

I.	Number of Instruction/Stimulus Screens	86
	Display Generation Time	5 h
II.	Experiment Description Length	250 statements
	Coding and Keypunching Time	6 h
III.	Testing and Debugging Experiment Description	3 h
	Total	14 h

In order to provide accurate subject response latencies, the time of onset of stimulus must be known. The device described above provides this facility by generating an input signal on one of the eight bits of the input line associated with a particular terminal. Two other bits of the digital line are used by the experiment monitor for special experiment station functions: aborting the current experiment and requesting input of information from the display. The remaining bits of the input line are used for five subject response buttons.

CONCLUSIONS

The "total system" concept has been received enthusiastically by the experimenters for whom the system was primarily designed. Experimenters have been greatly impressed by the rapidity with which they can implement, run, and analyze data for an experiment using the system. Experiment development has been, as we expected, largely dependent on the data entry tasks (generating the Display File and keypunching the experiment description). An example of development time is presented in Table 1.

DTES has been in operation for several months at the writing of this paper. During this time it has become apparent that there are serious impediments to the generalization of the system to other types of experiments. This problem is due to the design of the Experiment Description Language and the interpreter. These modules were intentionally designed to allow implementation of the specific class of experiments mentioned earlier. However, experimenters are now anxious to extend the algorithmic power of the description language of the system in order to implement experiments that are ideally suited for the system, but which cannot be conveniently expressed in EDL. One possible solution that is currently being evaluated by the authors is substituting an existing higher level programming language for the Experiment Description Language and the entire real-time system. The hope is that the other elements of DTES (Display Edit, Data Recovery, and Data Analysis programs) can be "wrapped" around a powerful programming system such as EXTENDED SCAT (Polson & Campbell, 1972). This would create a second level "total system" for those users more familiar with programming. The total extent of the necessary additions to the existing SCAT system to permit this is not known at this time. However, it is presently felt that a relatively small effort would be required to accomplish this goal.

APPENDIX I

EXPERIMENT DESCRIPTION LANGUAGE SYNTAX

Often there are references to optional arguments in a statement. These optional arguments are designated by putting them in square brackets ([ ]).

Initialization Commands

These commands are used to initialize the interpreter and *must* be the first four commands on the file. However, they can be in any order.

CONDITION cccc
DISPLAY FILE filename, area
DATA FILE filename, area
EXPERIMENTER cccc.....cc

The CONDITION argument is any four alphanumeric characters which the user chooses to identify his experiment. The EXPERIMENTER statement argument is a character string (1-20 characters) which will identify the experimenter.

Pattern Statement

PATTERN cc...c

The PATTERN statement argument is a character string (1-20 characters). Inclusion of this statement in the experiment description deck signifies that the preliminary statistical analysis report is to be included with the Data Disposition Statement at data recovery time. Only those data records which match the pattern argument are included for the analysis.

End Statement

END CONDITION

This statement defines the end of the experiment description deck and terminates compilation.

Block Definition Statements

Each block has a unique label from 1 to 8 characters in length. This label must appear in the DEFINE and END statement for each block. Blocks can be nested to an arbitrary depth limited only by the current configuration of the interpreter. Otherwise they follow the syntax (but not the logic) of ALGOL type languages. The DEFINE and END statements are written as follows.

DEFINE cc...c
END cc...c

Code Execution: The Execute Statements

Much of the utility of this language comes from the block structure and the way blocks of code are executed. A block of code is treated like a FORTRAN

subroutine. It will not be executed until it is referenced ("called") in an EXECUTE statement or referenced by a feedback label in a "SHOW UNTIL" statement. The EXECUTE statement can appear in any block at any depth. The EXECUTE statement has two forms:

EXECUTE $label_1$, $label_2$ ...., $label_N$

This statement executes the named blocks in the order of their appearance from left to right.

EXECUTE m OF $label_1$, $label_2$ ...., $label_N$

This statement will permute the list of block names and execute the first m blocks on the list, where $m \leqslant N$. The EXECUTE statements function like a series of FORTRAN subroutine calls.

Counter Operations

There are four counters available which are included with each data record generated. They can be set to zero or incremented by 1. The counter operation statements follow:

RESET i INCREMENT i

where i is an integer from 1 to 4. Counters may be reset or incremented at any block level.

Wait Statement

The user can specify a wait for a specified response or a specified time interval. The response can be on one of five response buttons. These commands have the form:

WAIT FOR t

where t is an integer or real number specified in units of seconds.

WAIT FOR REMAINDER OF t

where t is a time interval (expressed in the format just described) which specifies a delay period from the last onset of stimulus time rather than the current time. This command permits the setting of a minimum intertrial interval.

WAIT UNTIL R1,... R5

where the argument list contains 1 to 5 arguments of the form Ri in any order. The display screen will remain blank during the pause caused by the WAIT command.

Show Statement

These commands cause a screen image to be copied from the display file to the display screen. The 'r' in the commands refers to the display file record number. The optional '(S)' suffix on the record number means that the screen is to be superimposed on the current screen.

The SHOW statements are of the form:

SHOW r [(S)] FOR t

where t is an integer or real number in time units of seconds, used to time the duration of the display, and

$$\text{SHOW r } \left[\text{(S)}\right] \text{ UNTIL R1} \left[\begin{matrix}\text{(C)}\\\text{(R)}\end{matrix}\right] \text{[=label]} \text{,...,R5} \left[\begin{matrix}\text{(C)}\\\text{(R)}\end{matrix}\right]$$

[= label] [SID = XX...X]

The Ri refer to one of five response buttons and the optional flags '(C)' or '(R)' indicate a *correct* or a *reset* response. A reset response will cause the current flow of execution to be abandoned and the block containing the SHOW statement to be restarted if the flagged response matches the subject's response.

The optional field ' = label' indicates a block to be executed if the associated response is the response made by the subject. This feature can be used to provide feedback for the subject. Note that the ' = label' field can be used with the optional '(C)' flag and that it can be used alone (without '(C)' or '(R)' but that it will not work with the reset option '(R)'. This is due to the fact that the reset flag unconditionally restarts the current block.

The optional 'SID = ...' field is a 1 to 20 alphanumeric field that is used to flag the data generated. The only time data is output is whin a 'SID = ' field is present on a SHOW UNTIL statement.

REFERENCE NOTE

Lewis, J. L., Osgood, G. W., & Herbert, J. J. Pleiades: Real-time-sharing control in the behavioral science laboratory. Unpublished.

REFERENCES

Bailey, D. E. & Polson, P. G. Real-time computers in psychology at the University of Colorado. American Psychologist, 1975.

McLean, R. S. PSYCHOL: A computer language for experimentation. Behavior Research Methods & Instrumentation, 1969, 1, 323-328.

Polson, P. G. & Campbell, G. D. EXTENDED SCAT. Behavioral Science. 1972, 17, 558-565.

Polson, P. G. SCAT: Design criterion and software. Behavior Research Methods & Instrumentation, 1973, 5, 241-244.

Restle, F., & Schaffer, W. D. Monitoring experimental results. Behavior Research Methods & Instrumentation, 1974, 6, 262-266.

Waite, W. M. A language independent macro processor. Communications of the Association for Computing Machinery, 1967, 10 (July, 1967), 433.

Wood, R. W., Settle, W. F., & Weill, B. Interfacing the experimenter to the computer: Languages for psychologists. American Psychologist, 1975, in press.