Behavior Research Methods, Instruments, & Computers
1995, 27 (4), 516-519

Software Review

Experimental RunTime System: Software for developing and running reaction time experiments on IBM-compatible PCs

ADDIE DUTTA
Rice University, Houston, Texas

The Experimental RunTime System (ERTS) is a recent addition to the small collection of commercial software packages available for writing and running reaction time experiments on IBM-compatible PCs. Experiments are written using any text editor and a relatively small set of ERTS commands. Millisecond timing is provided by a software timer with no additional hardware requirements. All displays are in high-resolution VGA graphics. The Creative Labs Soundblaster card is supported in 16-bit mode, but is not required for simple tone generation. Voice-key support is available via the Soundblaster card. Keyboard, mouse, and external keys may be used as reaction keys, and the mouse or a joystick may be used as a tracking device. The stimulus-centered design and powerful display control commands of ERTS make it appropriate for developing a wide range of trial-based experiments.

If you are currently writing your reaction time experiments for IBM-compatible PCs in a conventional programming language or have difficulties with your current experiment-authoring software, you have probably wished for an alternative method of getting the job of authoring experiments done. The *Experimental RunTime System* (ERTS; Beringer, 1993; Iwanek, 1994) is designed for researchers who wish to program experiments and collect data without recourse to a general-purpose programming language. ERTS has been used in over 80 academic, industrial, and government laboratories in Europe and the United States since 1988, and it was used to generate the experiments conducted on several space missions (Manzey, Lorenz, Schiewe, Finell, & Thiele, 1993). The battery of sample experiments included with ERTS gives a good indication of its usefulness and provides templates for the generation of new experiments. Among the experiments included in the sample battery are: visual search, Stroop, memory search, pursuit tracking, tracking with concurrent memory search, and vigilance tasks.

Perhaps the strongest feature of ERTS is the way that text and graphic objects are handled; researchers will appreciate the ease with which stimuli are defined and positioned. The software is somewhat difficult to learn, but the intuitive structure of the experimental language and excellent debugging facilities compensate for the lack of tutorials. In this review, I give an overview of the structure of an ERTS experiment and some special features of the software, and document some of the difficulties I encountered in learning the software. Although I include a few hints on where to look for information and discussion of many aspects of the software, this review does not provide either a tutorial or a complete description of ERTS.

AUTHORING ERTS EXPERIMENTS

ERTS experiments are written using any text editor or word processor. This has the advantage of allowing maximum user flexibility in editing code without having to go through a scripted format or an excessive number of templates or forms. But, as with other software packages, greater flexibility and freedom come at a price of longer learning times and, on occasion, time-consuming error detection. Because experiments are programmed in a text editor, no errors are detected at the time of typing. This differs from the main competitor of ERTS, MicroExperimental Laboratory (MEL, Version 1; Schneider, 1988), in which at least some errors are detected when form template slots are filled in or when the program is generated. However, each component of an ERTS experiment (stimulus definitions, trial structure, etc.) can be tested individually. The different options for testing parts of ERTS code are convenient to use because one part of the code can be tested before other parts are

I thank Bill Maki, Jörg Beringer, and an anonymous reviewer for helpful comments on an earlier version of the manuscript. Correspondence concerning this article should be addressed to A. Dutta, Psychology Department, P.O. Box 1892, Rice University, Houston, TX 77251 (e-mail: duttaa@rice.edu).

written. For example, it is possible to view all stimuli and their positions without programming a trial structure. This excellent feature, not available in MEL, compensates in part for the lack of automatic error detection and the sometimes cryptic error messages.

Structure of the ERTS Experiment

Programming in ERTS is best described as "bottom-up." Stimuli, screen positions, responses, and even trials are defined as data types and form the basic elements for trial blocks and sessions. This experiment structure and the wide range of stimulus types, flexibility in trial definition, and wide choice of response modes make it possible to create experiments ranging from same–different matching, to rapid serial visual presentation, to cross-modal reaction time tasks.

Definition of stimuli and responses. The first step in creating an experiment is to define all of the text and stimulus "objects," such as instructions, stimuli, and feedback. Several fonts are supported and, like all visual information in ERTS, are displayed in high-resolution graphics mode. Such complete utilization of the high-resolution graphics mode is a nice feature, in that a high degree of control over text (such as the size and spacing of letters) and integration of text and graphics are facilitated. ERTS will even rotate text or images as specified by the programmer. Included with ERTS are special font resource files that allow characters to be displayed in bitmap fonts in sizes up to 60 pt. Except for the default font, each font to be used must be declared within the ERTS experiment.

There are several options for creating graphics. Very simple objects can be defined pixel by pixel within the experiment code, PCX files can be converted into an ERTS-readable format using the PCX2GRA utility (included in the ERTS package), or a special DUMPSCR utility (also included) can be used to capture screen images much like the "print screen" option does. Although regions of the screen can be selected to create a desired image size, the images themselves cannot be scaled by ERTS. All graphics are converted to black and white by ERTS, but the resulting files can be displayed in any combination of foreground and background color. Colors are defined by the user, using one of three color models (proportion of red, green, and blue; hue, saturation, and value; or red, green, and blue intensity). Virtually any number of colors may be defined, but only three colors (in addition to the background color) currently can be displayed on a given screen.[1]

Positions are defined independently of text and graphics objects, which facilitates the efficient definition of stimuli. Pictures are displayed centered on the *x*/*y* coordinates of the positions (coordinates are in tenths of millimeters and the coordinate system can be adjusted to fit individual monitor sizes), and a wide variety of options for defining positions is available. Positions can be defined as *points* (given in vertical and horizontal coordinates), *lines*, *vectors*, *circles*, or *matrices*. One or more positions may be defined with the point option. To use the line, vector, or circle options, the user specifies the location and size of the display (e.g., a 30-mm circle centered on the screen) and the number of objects to be located within it. ERTS then determines the coordinates to be used for equidistant display of items. The final option for positioning objects, the matrix command, generates a regular grid of locations that can be indexed by column and row. All of these position definitions can be used by built-in random processes such as filling empty locations with pictures randomly selected from a set or positioning pictures on random locations.

Another potential advantage of ERTS over existing software packages is the flexibility with which auditory stimuli may be defined if the user has a Soundblaster Pro sound card (Creative Labs) installed. Without a sound card, auditory stimuli are limited to sequences of tones specified only by frequency and duration. With a sound card, virtually any sound or speech (VOC files or MIDI notes) can be presented, as allowed by memory and time constraints. One touted advantage of using VOC files is that they can be played in the background (without interrupting the timing of other events) or can be synchronized with other events at voice markers or the end of the voice; however, I encountered some difficulties in getting these options to work.

In addition to specifying the stimuli and stimulus positions, the programmer must specify the allowable response keys whenever more than one key is allowed as a response. Either keyboard or external keys (including mouse buttons) can be used to collect manual responses and either key depressions or key releases can be registered. Changing the response mode in an experiment often requires only the redefinition of response keys. A special feature of ERTS is that *no response* (as in a go/no-go paradigm) can also be specified as the correct response alternative. The manuals mention that voice keys (via the soundcard and a microphone) are also supported. Advertised, but not included in the review version of the software, is an EXKEY microprocessor logic,[2] which allows 8 (with make and release registration) or 16 (make registration only) external keys to be used for more precise timing, especially of multiple or simultaneous keypresses, than is possible with keyboard or mouse keys.

Trial definition. Having declared all of the text, pictures, tones, positions, and response keys that will be used in the experiment, the next step is to define the trial sequences. Trial definitions specify the timing of events and response collection and feedback to be given following responses, as well as the fonts to be used and the tones, pictures, or text to be displayed and their colors and positions. It should be noted that responses can also be collected continuously across trials, making it possible to conduct experiments such as vigilance tasks in which responses can occur at any time.

Up to 25 different trial commands are used to describe the series of events that make up one trial. For example, the *SP* (Show Picture) command displays a predefined text or graphic object in the position and color indicated,

for a specified amount of time. Relatively complex displays can be created from simpler elements using the *concatenate* (join all picture elements before displaying the composite picture) or *add* (add new elements to the screen without erasing displayed elements) options. Stimuli can be presented for a predefined time, until a response is registered, or until the maximum amount of time allowed for a trial is reached. Variable arguments may be used where appropriate in trial definitions (e.g., for variable stimuli, positions, colors, and correct responses), which allows the same trial structure to be used to present a range of trial types. Values for the variables are then specified within the block or session definition.

A variety of randomization procedures can be used within the trial definition, including defining a set of stimuli and drawing from the set with or without replacement. Alternatively, each possible trial type can be listed in the block definition (described below) and then executed in random order.

ERTS has a built-in trial command for response-dependent feedback. The desired feedback (visual or auditory message) is specified by the user, then the feedback option is invoked to link each feedback type with the nature of the response (e.g., correct, incorrect, too fast, or too slow). The maximum allowable reaction time is 30 sec. It is also possible to specify a response without recording reaction time.

Block definition. Although trials may be executed from different levels of an experiment, reaction times are collected only when the block command is used to run trials. The block definition lists all of the trials that should be run within a given block of the experiment. Conceptually, a block is one replication of a factorial design. For example, if four trials each of three different types of trials were to be run in a block, the block would be defined as having a 3×4 structure and the data file would be coded accordingly. If only one type of trial is to be executed, only one number need be specified. The numbers specified in the block definition must match the number of trials defined. So, in the example above, ERTS would expect 12 trial definitions to be listed.

For each trial, the number of times to execute the trial, trial name, correct responses (where appropriate), and any variable values are listed. For example, if an experimental trial requires that one of four stimuli appear in one of three positions, the trial specification within the block statement would give the name of the picture and position to be used. Additional column headings for coding the data file can also be specified within the trial statement.

Session definition. The session is the highest level in the experiment, and only one session may be specified. From this level, either trials or blocks may be executed and default values for foreground color, background color, font type, and font size may be set. Blocks are run by specifying the name of the block, the number of trials to be run (which must be a multiple of the minimum block length), the number of warm-up trials to be run (i.e., trials for which no data are to be collected), and the randomization procedures to be followed. Blocks can be terminated by session commands that define the maximum number of errors or amount of time allowed for a given block. As mentioned earlier, although trials can be executed directly from the session level, no data will be collected for these trials. Response-dependent branching is allowed at the session level, but is limited to conditional jumps from the current location in the program to a destination label. Finally, the experimenter can easily program the experiment to collect additional subject data (entered by the experimenter) at the beginning of the experiment.

Data analysis. The data analysis software (ERTSCODE) is rudimentary in comparison to other available packages, such as MEL. However, all of the functions needed to prepare the data for transfer to statistical packages are provided. Also, several options exist for processing tracking data.

Experiment Development

The sample experiments included with the ERTS software can be used as templates for generating new experiments. As mentioned above, each aspect (stimuli, positions, trial structure) can be tested individually. In addition, the temporal resolution of the experiment can be changed for testing purposes and the experiment can be run in a noninteractive mode so that no subject responses are required. A very useful feature, and one that is lacking in MEL, is that experiments can be aborted after only a portion of trials have been run and results will be stored up to that point.

TIMING CONSIDERATIONS

ERTS offers many features to control the timing of experimental events. For example, data are written to the disk only at the end of blocks to ensure accurate timing across trials, although this means that timing between blocks cannot be guaranteed to be accurate. Not all details of the routines used for timing are described in the ERTS documentation (see Beringer, 1992, for more information), but the manuals that come with ERTS point out various problems that could arise. For example, echoing to the screen text that is typed by the subject might delay the following events, as will the presentation of some types of sound files. A special option (/W) can be used during experiment development to check the accuracy of the times defined in the experiment. Finally, a screen synchronization option will round display intervals to the next vertical retrace, so the monitor refresh rate should be checked using the ERTS option provided for this.

The author has provided a built-in Sternberg memory search task that can be used as a secondary task with tracking tasks, but this is the only true dual-task option available. However, multiple stimulus presentations and response collections can be defined within a trial. Thus, it is possible to present two stimuli, separated by a stim-

ulus onset interval, and to collect a response to each stimulus. The second reaction time must then be corrected later to take into account the delay between the two stimuli.

A feature of ERTS that deserves special mention is its use of screen paging between different video memory areas to switch graphics images within one screen refresh cycle. Such pseudotachistoscopic image switching is realized by a special *Intelligent PreLoad* algorithm which is applied automatically whenever the next screen to be presented is predefined (a case in which the next image is not predefined would be when visual feedback was dependent on the subject response). Thus, image switching is normally available within one screen refresh cycle regardless of the complexity of the images.

LEARNING TO PROGRAM IN ERTS

To learn ERTS in a reasonable amount of time (less than a week), you should have a working knowledge of DOS and facility with a good text editor. Previous experience in programming would be helpful, especially with regard to formulating the desired experiment in terms of programmable "objects." A very brief overview in the user's manual gives the basic structure of the ERTS experiment. However, individual commands and their uses are not adequately explained in the overview, making it of limited use except as an "advance organizer" for a careful reading of the reference manual. Since tutorials for learning ERTS are virtually nonexistent, some dedication is required to learn the language. Only the most minimal instruction is given, which gives the learner the sense of discovery whenever anything works—but most of us would settle for less excitement and faster learning times. Fortunately, e-mail support from the developer is prompt and to the point. Finally, note that ERTS is bilingual; the list of reserved words extends to German translations.

The lack of user-friendly devices such as fill-in-the-blank forms or templates has a positive aspect in that questions never arise regarding how to integrate user-written ERTS code with such automatically written code. Thus, although other packages, such as MEL, may provide an initial learning advantage, it is likely that many of the same problems would be confronted eventually as more ambitious programming projects are attempted. Experienced programmers will appreciate the high degree of control possible in accessing peripheral devices and controlling screen images, as well as the efficiency of the randomization procedures.

SUMMARY

Although the difficulty of finding information does not detract from the functionality of ERTS (unless one is trying to track down the meaning of error messages, for which there are no listings in the manuals), it may dissuade some people from putting in the time necessary to learn the program. This would be a shame. Despite the shortcomings in documentation and tutorials, ERTS offers some great features. No other software that I know of allows for the generation of both reaction time and tracking experiments, and the location-independent object definition is both intuitive and flexible. In addition, some of the special features, such as automatic filling of locations with distractor items in visual search paradigms, will make the acquisition of ERTS especially worthwhile for researchers who use such paradigms.

Availability

Experimental RunTime System. Jörg Beringer, 1993, Berisoft Cooperation, Wildenbruchstr. 49, 60431 Frankfurt am Main, Germany. $850 for two computers (includes shipping). Double the number of computers currently licensed can be added for additional payments of $850 Demonstration disk available upon request.

REFERENCES

BERINGER, J. (1992). Timing accuracy of mouse response registration on the IBM microcomputer family. *Behavior Research Methods, Instruments, & Computers*, **24**, 486-490.

BERINGER, J. (1993). Entwurf einer Anwendersprache zur Steuerung psychologischer Reaktionszeitexperimente. *Europäische Hochschulschriften* (Reihe XLI, *Informatik*). Frankfurt: Peter Lang.

IWANEK, R. (1994, November). Review of the Experimental RunTime System. *Psychology Software News*, **5**, 65-67. (CTI Centre for Psychology, University of York, U.K.)

MANZEY, D., LORENZ, D., SCHIEWE, A., FINELL, G., & THIELE, G. (1993). Behavioral aspects of human adaptation to space: Analyses of cognitive and psychomotor performance in space during an 8-day space mission. *Clinical Investigator*, **71**, 725-731.

SCHNEIDER, W. (1988). Micro Experimental Laboratory: An integrated system for IBM PC compatibles. *Behavior Research Methods, Instruments, & Computers*, **20**, 206-217.

NOTES

1. The vender has announced the availability of a 256-color version of ERTS capable of presenting 256-color scanned images. This product can read BMP, ICO, and RLE graphics files directly. The demonstration program for this product requires 2 MB of video RAM.

2. The EXKEY logic can be purchased for $380. Keypads can also be purchased from Berisoft, or user-supplied keys can be connected.

(Manuscript received June 3, 1994;
revision accepted for publication September 21, 1994.)