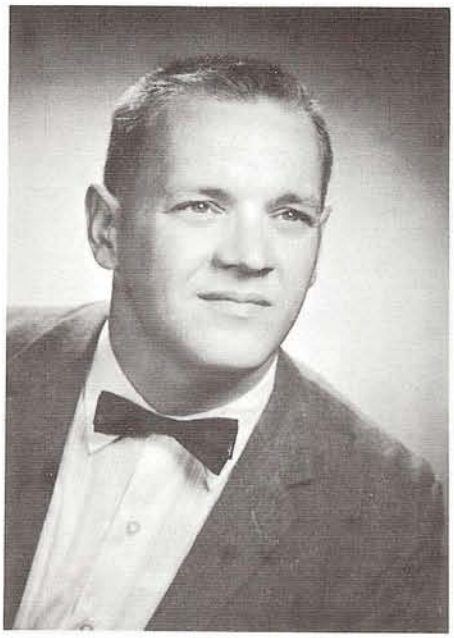
Developing Real-Time Systems

by George R. Trimble, Jr.



George R. Trimble, Jr., obtained a B.A. from St. John's College in 1948 and an M.A. in mathematics from the University of Delaware in 1951. Formerly with the Computing Laboratory of the Ballistics Research Laboratories and a senior staff member of the Applied Science Div. of IBM. Mr. Trimble joined CUC in 1956. Utilizing experience on a multitude of computer systems, he now performs analyses and supervises major programs for many applications, primarily in real-time and systems programming. His present position with CUC is Corporate Technical Director of Systems Programming.

Introduction

Any reasonably complex real-time system necessarily involves tremendous problems of coordination and control. The very nature of such an enterprise requires the utmost in accuracy and reliability of results. The analysis of each phase of the problem, as well as the analysis of the overall system must be well planned, in order that the integration of the various sub-systems can be done effectively and efficiently. This planning must permit effective control over the progress of each phase of the project development. Documentation procedures must be clearly defined and rigidly adhered to in order to permit this necessary control.

The first step in designing the computer programs must be a precise definition and thorough analysis of all the physical and human engineering factors. Based on the requirements derived from this analysis, an overall system design can be developed for utilizing the computer which will greatly enhance the functions which must be performed.

The system design requires several different types of aids. Programs to implement the required functions must be developed and a programming system which is oriented toward the ease of development of these programs is desirable. During the debugging stage, various programming aids are required to assist in checking out parts of the programs and integrating them together. During system integration and system testing, large volumes of input data are required. Methods for generating input data, recording intermediate and final results, and analyzing the results of a test are necessary.

The requirement that the system operate in real-time imposes restrictions and creates problems which do not occur in normal data processing. These problems are further complicated by the fact that the computations must be performed according to a priority scheme.

The following sections outline programming and analysis areas which must be considered in the development of a real-time system.

CONTROL PROGRAM

The Control Program exercises supervisory control over all the computing equipment during a run. It must perform several tasks to handle different situations that may arise and assist programmers in efficient debugging of their programs. Included among these tasks are:

PROCESSING INPUT-OUTPUT REQUESTS - The input-output devices may be activated by the program on request of the device itself, or on a time sequence basis. Requests for input-output by the program are transmitted via the control program to the input-output program. If necessary, these requests are stored in queues until they can be accepted by the requested device. The Control Program must be capable of determining the order in which input-outputs must be processed (the priority problem.) Input signals from input-output devices are analyzed by the Control Program to determine the action which must be taken. These interrupts can be initiated by input-output devices to indicate presence of a specific condition or can be initiated by an operator at a console.

DETERMINING PRIORITY OF INPUTS AND OUTPUTS - The rate at which input data is supplied by, or output data sent to an input-output device is frequently fixed by the characteristics of the device. Higher speed devices must be serviced quickly or the information may be lost. The Control Program must determine the processing order when more than one device requires service, to assure that high priority devices receive the needed service in time.

Priorities may be assigned to some devices on the basis of the importance of information supplied by the device. For example, an input signal to initiate some emergency action may take precedence over any other input. The processing of priorities must consider such externally assigned priorities as well as the natural ones arising from equipment characteristics.

PROCESSING ERROR SIGNALS - Various types of errors can occur during the debugging or operation of the system. Some of these will be program errors in which the programmer may wish the job to be terminated, or may indicate some corrective action to be performed. Errors in input-output operations must be processed to cause the operation to be re-initiated, some alternative device to be used, or other corrective action to be taken. Other types of equipment errors may require that the program be restarted at an earlier point. The Control Program must process these various types of errors and initiate the proper corrective action for the specific type of error which has occurred.

CONTROLLING RESTARTS - In some cases, errors detected result in the program being restarted at an earlier point. Various types of restarts will be required depending upon the nature of the error. The Control Program must analyze the type of error to determine what type of restart is required. It is necessary that critical information be saved in order that a restart can take place. The Control Program must periodically save this information and keep track of the control points which determine how far back processing must be initiated, depending upon the data saved and the nature of the error.

CONTROLLING INPUTS - Since the system will be built up in stages, there will be periods during which hardware input devices will not be available. Until they are available, it is necessary to simulate these devices. The Control Program must provide the facilities for selecting whether an input device is actually connected to the system or if input data for that device is to be entered via a simulation process.

In a run where a device is being simulated, it is necessary to specify the conditions under which the simulated data is to be entered. For example, some data may be required at specific times, in which case a record of simulated input data must be read in when that simulation time is reached in the program operation. Other input data may be required when some condition is reached in the program. This latter case may be the entry of sub-system data when the operator at a console presses a button.

CONTROLLING DATA RECORDING - During the program integration and system testing phases, it will be necessary to examine relatively large amounts of the intermediate and final results generated by the system. It is necessary to record this data on some media, presumably magnetic tape, so that it can be analyzed off-line. The recording of this data must be under control of the Control Program.

COMMUNICATING WITH OPERATOR - Another function of the Control Program is to communicate with the operator so that he can be informed of the status of the system and of any actions he must take, such as changing tape, loading new programs, etc. It also permits the operator to indicate to the system any changes he wishes to occur.

LOGGING - To facilitate maintenance of the system, the Control Program should log significant events. Such information as data on types of errors processed and restart procedures activated is useful to the maintenance engineers to locate equipment difficulties.

Dynamic Simulation Programs

Dynamic Simulation refers to making the system go through a run, but not necessarily in real-time and not necessarily with the digital computer connected to all external devices. Real time programs require Dynamic

Simulation techniques to test the system under conditions which are as close to its anticipated operational environment as possible. Frequently, large volumes of data must be generated as input to replace the missing hardware. Large volumes of output data may be generated and must be analyzed to evaluate a particular test.

Dynamic Simulation is useful for testing, and also facilitates the incorporation of actual hardware at a later stage. By testing the programs using data normally taken from a device (which simulates inputs from the device), program errors, owing to incorrect processing, are minimized or possibly eliminated. Thus when the actual hardware is connected, the difficulties encountered and errors have a high probability of being due to the hardware and its inter-connections, rather than incorrect processing by the digital computer program.

Several different sub-systems are potentially useful for Dynamic Simulation. These are described in the following:

DATA TRANSLATION PROGRAMS - Data generated by the device to be simulated must follow a specific format for entry into the computer. It may be inconvenient or awkward for the programmer to prepare data for testing his program, which is in the exact format produced by the particular device. Data Translation Programs convert from a convenient programmer-oriented data format to the actual machine format. An example would be an input device which generates binary data. The corresponding Data Translation Program would convert from decimal to binary and prepare magnetic tape or punched cards in a binary format for entry into the computer at program execution time.

The Data Translation Programs are primarily used to aid in preparing small amounts of data during sub-programs debugging phases. They are of little value when it is necessary to prepare large amounts of data.

DATA GENERATION PROGRAMS - Large scale system tests require relatively large volumes of input data. For example, the simulation of radar returns from a target for a thirty minute period with a ten per second sampling rate would require the preparation of 18,000 pieces of data. A Data Generation Program could generate the required radar returns based on the projected course of the target. The programmer would specify the initial conditions and course parameters and the Data Generation would generate position data corresponding to the ten per second rate and derive numerical values for the radar returns based on these positions.

The Data Generation Programs would be useful not only for debugging the programs, but could be of value during a run in which it is not desirable to actually use real equipment. The generated data can be much more precisely controlled by the program or systems tester to fill the precise needs of his specific test run.

Data Generation Programs could also insert noise in the generated data which follow a prescribed frequency distribution. Thus, in the case of radar returns, the program could generate exact returns and then augment them by errors to give a more realistic simulation of what would actually take place.

During the program integration, it is possible and indeed probable that some sub-programs will be ready for debugging prior to the completion of debugging of other programs which produce data to be processed by them. The Data Generation and Translation Programs can be used to create these intermediate results so that individual sub-programs can be more fully debugged before they are integrated with their related sub-programs.

SIMULATED DATA INPUTS - The data created by the Data Translation and Generation Programs must be entered into the system at the time it is needed. As indicated above, the Control Program controls the reading of simulated input data. It is necessary, however, to specify the conditions under which the simulated data will be entered. This can be on a time basis under control of the computer program or on an event-basis, depending upon occurrence of some specific conditions during the run.

Special System Simulation

In some portions of the system, it is possible to predict what the output will be without going through the actual program. Using the example of radar tracking again, the course prepared by the programmer will generate data which will simulate the relative positions of target and the radar. At some specific point on the course, an event should occur, for example, separation of booster and second stage in tracking a rocket. The position and time at which this separation should occur can be precisely predicted and the system should take definite action at the point when the run is made.

In the example given above, it is not necessary to perform elaborate computations to determine the position that the significant event (booster separation) takes place. Other situations, however, may require a small computer program to determine what the results of the run should be.

Inputs to the special system simulation programs may be generated data, which are also used as inputs during normal run or they may be data recorded during an actual run.

The special system simulations can include calculations to determine time delays for pertinent portions of the system in order to determine actual vs theoretical timing relationships. For example, if a special system simulation run is made in which only one sub-system is simulated, the total time required by the sub-system can be determined. Following this by an actual run, the actual time is obtained which will include delays due to processing data from other sub-systems. Comparison of these two runs will reveal the extent of time delay of the sub-system being examined due to the operation of it in conjunction with other sub-systems.

Data Recording Programs

In order to evaluate a test or simulation run it is necessary that relatively large volumes of data be recorded for off-line analysis. As indicated above, the recording of this data is under control of the Control Program. The data recording programs, however, must have considerable flexibility in the selection and mode of recording of the selected data. On the other hand, recording all intermediate and final results gives too much data so that a major portion of available time will be spent simply recording data. Means must be provided whereby the program can select specific subsets of data to be recorded. Further selectivity can be provided by conditional recording. For example, an output message to a specific device can be recorded only if it differs from what the programmer has predicted it to be.

The format of recorded data will generally be machine oriented since it is not desirable to use time during a simulation run to change the format for easier consumption by the programmer. Further selection and editing of the recorded data must be done by the Data Analysis Programs.

Data Analysis Programs

During a run considerable data must be recorded, as indicated above, in order to evaluate the results of the run. Several auxiliary programs are required to process this recorded data to assist in evaluating the run. These various analysis programs are indicated in the paragraphs below:

SELECTED OUTPUT DATA - Although the selection process has already been performed by the Data Recording Programs, the particular analysis to be performed may require a subset of all of the data which has been recorded. The first step in the analysis is to select this subset of data which is to be further processed by the Data Analysis Programs.

SORT OUTPUT DATA - To consolidate groups of related data, it may be necessary to sort the selected data to assist in the manual verification. Thus, the Data Anaysis Programs may include several types of sort programs.

CORRELATE OUTPUTS - For some tests, the results expected from the test can be predicted beforehand. It is desirable to correlate these results with the actual results obtained during the test run. These programs are effectively, merge programs for the various types of correlations required.

STATISTICAL PROCESSING - Some analysis programs require computing statistical data from the selected output data. Such things are means, standard deviations, correlations, power spectra, etc.

Timing Studies

The time at which events occur or delays in processing information at various points in the system is often important. This implies that the data recording programs have recorded time, along with the other information, so that timing studies can be made. The example of determining time delays described in the subprograms for special system simulation illustrates one of the types of study it may be desirable to perform.

Editing

The basic data recorded by the Data Recording Programs is generally highly machine oriented and, therefore, cannot be easily interpreted directly. Editing programs perform the conversions from machine code (for example, binary) to a form readily interpretable by the analyst. This would also include insertion of blanks between fields, puncuation in fields, and column headings to identify the data.

Another form of editing which may be desirable is editing of recorded data and generated tapes to abstract information to be entered as inputs into the System Simulation Programs.

Input-Output System

The input-output system is obviously one of the most important portions of the computer program. It must provide the capability of communicating with all of the standard computer input-output equipment. The input-output system must be integrated with the compiler used as much as possible, in order to facilitate writing the the programs. It must also have considerable flexibility to be able to handle the variety of special devices which the system will involve.

The analyst must consider the accuracy of inputs and outputs, and the frequency of the signals to be sampled in order to determine the sampling rate and scaling input-outputs. In some cases, the operation of the device must be controlled by the computer, while in others the device operates independently and the computer must conform to the device. Detailed analysis of the problem and equipment to be used must be made to answer these questions.

Reliabilty

Reliability data can be gathered as a by-product of

many runs, but it is easier to prepare specific tests designed to determine the system's reliability.

Data Generation Programs, which are similar or identical to the Data Generation Programs described earlier, can be used to generate specific types of data which will aid in the reliability evaluation. A simple example of the types of data which can be generated would be repetitive patterns of data which are easy to analyze.

Data recording is also useful for the reliability evaluations. Recording and analysis of intermediate results generated is necessary to determine whether the input has been properly processed.

Analysis of the data recorded by the Data Recording Programs could also be performed automatically in some cases rather than depending on a human being to study the output to detect discrepancies between computed and expected outputs. Data Analysis Programs to compare expected results with actual results can be used to automatically detect any descrepancies and give a summary report listing them.

Adaptation Testing

Another type of testing which must be performed before the programs can be accepted is adaptation testing. Some of the routines can be written as general purpose routines, adaptable to various uses. By specifying what the routines parameters are, these routines can be adapted to specific uses. For example, a routine can be generated which will simulate the effects on the system of a rocket motor. This program can be adapted to specific rocket motors by specifying the parameters of the motor. Obviously, there is loss of efficiency in making a general purpose program applicable to all motors, but if the characteristics of the motors encountered are reasonably similar, this approach is very effective. However, in order to assure that the system will work with all of the motors to be encountered, the subroutine must be tested using the parameter values which will adapt the subroutines to that motor.

Maintenance & Diagnostics

Standard programs exist for performing the maintenance and diagnostics of many commercially available computers. The principal program development in this area will be programs to test the special input-output devices which are connected to the system. These may include consoles, analog computers, data communication links, etc. The extent of these programs must be carefully controlled, since there could easily be a tendency to write diagnostics systems which contain more instructions than the application itself.

Programming Requirements

In previous sections a number of the areas where analysis and programming is required in connection with a major real time system were discussed. In the present section we discuss programs and programming requirements in more detail. A frequent method of proceeding with a new project is to take one phase of the problem and prepare specific specifications for the detailed computations, control program, input-output, linkage, etc., for that one phase, and then make every effort to get that one phase working on the equipment currently available.

The difficulty with this approach is that much of the work has to be repeated again and again as more phases are added to the simulation structure. (The advantage of this approach is that one has something tangible to show for one's effort in a relatively short time.) Should this approach be followed, there is a lot of work which should be done concurrently. By making a thorough analysis of the problems to be solved in relation to the equipment available, it is possible to write a number of general purpose programs which make programming debugging for simulation of specific phases easier.

The general programs are:

A Control Program
An Input-Output System
A Program Testing System

The set of programs will be called a programming system. With the help of a well-organized programming system, subsequent programming and testing efforts can be directed toward effective use of the real time system, rather than toward tedious, error-prone processes of manually preparing and analyzing large volumes of data.

In fact, the use of a programming system such as the one outlined is essential to the efficient, economical development of programs for any real time application. The programming system permits the flexibility and ease of use which are necessary to meet many problems which will arise. Data can be generated for program testing and simulation runs with a minimum of effort. The data generated is much less susceptible to errors than manually prepared data. In fact, it would not be practical to manually prepare the volumes of data which are necessary for thorough testing of the system.

Dynamic simulation techniques provide greater assurance that the programs are operating correctly when actual hardware is integrated into the system. This minimizes the problem of trying to determine whether erroneous results are caused by hardware malfunctions or bugs remaining in the programs.

Data analysis provides faster, more accurate, more meaningful evaluation of test and simulation runs. Thus, system evaluation is accomplished much more quickly than manual data analyses techniques would permit.

Control Program Components

The Control Program is entered under various trap and programmed transfers. It consists of the following:

Input-Output Processor
Error Processor
Priority Processor
Restart
Dynamic Simulation Inputs
Data Recording
Communications
Logging

The Control Program should be written modules, so that when certain functions are not needed, the associated instructions are not entered into the computer.

Program Testing System

The program testing system consists of the supporting programs required to assist in the program development. This system consists of the following programs:

Data Translation Programs
Data Generation Programs
Dynamic Simulation Programs
Special System Simulation Programs
Data Recording Programs
Data Analysis Programs

In addition to these programs, portions of the Control Program are required for controlling the programs during a test or simulation run. The functions performed by these programs have been discussed previously. In order to illustrate more clearly what these programs consist of, we describe one set, the Data Recording programs, in more detail in the next section.

Data Recording Programs

FUNCTION - To record specified input and specified intermediate and final results selectively during a dynamic simulation or actual run.
CONTROL - The programmer prepares cards which specify the data to be recorded. These cards are processed to generate tables which are referred to by the Control Program during a run, to determine whether the data is to be recorded or not.

The control cards specify the following parameters which define the characteristics of the data to be recorded:

Identification of item to be recorded
Conditions to be met for recording item

The item identification is established in several ways, depending on the nature of the item. Inputs from or outputs to a device can be identified by the name (channel, unit number) of the device. Data transmitted between sub-programs can be identified by the memory location of the data. To facilitate use of the Data Recording facilities by non-programming personnel, items should be identified symbolically whenever possible with pre-processors to convert from symbolic to absolute identification using tables and dictionaries generated during the assembly or compiling phase.

Conditions to be met for recording may include the following:

Unconditional recording of item
Record every nth item
Record only when item does not equal a specified value

RECORD FORMAT - Data selected for recording must be adequately identified so that it can be processed by the Data Analysis Programs to derive the information required for evaluation of the test run. Each record should contain the following:

Time item was recorded
Identification of item
Item

For efficiency of use of magnetic tape, groups of items should be blocked before written on tape. A buffer area must be provided with the Data Recording Program to enable this blocking of items.

Use Of The Program Testing System

The Program Testing System will be used in two ways during the course of a project. During the program development, debugging, and integration phases, it will be

used to aid in data preparation, program check-out, and system testing. When the system becomes operational and is being used for actual runs, parts of the Program Testing System will be useful to aid in data preparation and analysis of results of a run.

Some of the uses of the Program Testing System in the first of these processes are:

In debugging individual programs using manually prepared test data which is processed by the Translation Programs.

In preparing test data using Data Generation Programs Developed for that part of the problem.

By preparing control cards, Data Recording Programs selectively record data necessary for testing programs.

By preparing control cards, define real vs. simulated inputs for a test run.

In making test runs.

By preparing control cards, Data Analysis Programs select, sort process, and edit recorded data.

During an actual run, some uses of the Program Testing System are:

To prepare simulated input data (if any required) using Data Generation Programs.

For Data Recording.

For defining real vs. simulated inputs (if any) for a run.

In making an actual run.

For Data Analysis.

For processing recorded data using Special System Simulation Programs and Data Analysis Programs.

Evaluation of results of a run.

Summary

Development of large real-time systems requires the solution of problems which do not exist in standard data processing applications. Dynamic Simulation provides tools which greatly facilitate the debugging and systems testing of such a system. Since real-time systems are becoming more widely used, these techniques will become increasingly important.

CUC COMPUTER USAGE COMPANY, INC. 344 Main Street, Mount Kisco, N. Y. 10549