CERN - DATA HANDLING DIVISION
DD/72/26
The ERASME Group - Cern
October 1972

THE USE OF A STANDARD TIME SHARING OPERATING SYSTEM TO CONTROL A COMPLEX OF REAL-TIME PROCESSES

(To be presented by D. O. Williams at Datafair 73
Nottingham 10-12 April 1973)

DD-ag

INTRODUCTION

When faced with the problem of providing software control for a complex of real-time processes it is fairly common to construct a purpose-built operating system. This paper presents our experience of taking the opposite approach and using a standard manufacturer-supplied operating system as the basis of the software system.

Section 1 gives a short introduction to the field of bubble chambers and Section 2 considers the computing requirements of bubble chamber film analysis systems. In Section 3 we consider the general design philosophy of a new film analysis system, called ERASME, that is being constructed at CERN by the group to which we belong. Section 4 contains a discussion of the software requirements of the ERASME project and we then describe our current software implementation in Section 5, with a description of how we foresee our future development in Section 6. Section 7 contains a discussion of the problems involved in our approach and, finally, in Section 8, we present some conclusions.

1. BUBBLE CHAMBERS

Bubble chambers are one of the major sources of experimental data for high energy physicists. They rely on the fact that when charged nuclear particles pass through liquid hydrogen, for example, just as it is about to change phase and become gaseous then points on their path become preferred centres of bubble formation. The chamber is normally made sensitive by using a piston stroke to decrease the pressure applied to the liquid. The reverse stroke of the piston then increases the applied pressure and clears away any bubbles that were formed. The piston typically has a cycle time of a few seconds.

The method of using such a chamber is to inject a group of up to about twenty particles, referred to as the beam, into the chamber shortly after it has become sensitive. In most experiments, the particles in the beam have been selected so that they are all of the same type (proton, pi-meson, etc.) and are travelling in the same direction with the same momentum. Many of these particles will pass right through the chamber and in most experiments, being charged particles, they leave behind a characteristic track of bubbles. Some of the particles, however, will interact with a nucleus in the chamber liquid. This nucleus will be a proton if the chamber has been filled with liquid hydrogen. In such an interaction the beam particle will either be simply deflected or else a number of charged and neutral particles will be produced. The charged particles will leave tracks of bubbles as they move away from the interaction point while the neutral particles will remain invisible. Some times the particles leaving the interaction point decay or interact again. The group of processes that originate from one beam particle interaction is referred to as an 'event'.

In the bubble chamber enough time is allowed, after the entry of the particle beam, for the formation of bubbles and then several flashes and cameras are triggered in order to simultaneously photograph the bubbles produced. A strong constant magnetic field is applied throughout the chamber so that charged particles move with near-helical motion. Provided that the path of a particle has been seen by at least two cameras a three-dimensional description of the helical motion can be obtained. It is common, however, to require that the path is seen by at least three cameras in order to achieve good redundancy. A knowledge of the magnetic field then allows the momemtum of the particle to be computed.

The physicist is normally interested in only a certain subset of the events that occur when the bubble chamber is exposed to a given beam. For the selected events the physicist needs to obtain an accurate measure of the momentum and direction of motion of each particle, together with an identification of the particle type. The accuracy requirements, translated into terms of film measurements call for the centre of a track of bubbles to be determined with a precision of the order of one tenth of the diameter of a bubble image (i.e. a few microns).

With a cycle time of a few seconds a bubble chamber is able to produce several million photographs, each consisting of three or four stereo-views, every year. Because of the statistical nature of the mechanisms that interest the high energy physicist, he is only able to draw conclusions after studying large numbers of similar events. Experiments involving the analysis of one hundred thousand events are now common, and several groups of physicists are analysing up to one million events.

It is the combination of these large numbers with the requirement for considerable accuracy that has made the automation of bubble chamber film analysis (1) so productive.

2. BUBBLE CHAMBER FILM ANALYSIS SYSTEMS AND THE ASSOCIATED COMPUTING REQUIREMENTS

We distinguish four stages of bubble chamber film analysis.

In stage one ("scanning") the film is inspected frame by frame in order to find events that the physicist wishes to study. The selection is normally made on the basis of some rather simple topological criteria. Except in some rather rare cases this selection is performed most efficiently by using a trained operator to inspect a projection of each photograph in turn. This procedure has not, in general, been successfully automated.

In stage two ("measuring") a detailed and precise two-dimensional description of all events selected at the scanning stage has to be obtained. At least fifty point measurements are normally required to give a reasonable description of an event in one stereo-view. The number of measurements and the precision required make this task very fatiguing for an operator and therefore many more or less automatic systems have been developed.

Most of these systems rely on the fact that when a beam of light is swept across the image of a line of bubbles then the intensity of the light transmitted through the film is modulated. If the motion of the light beam is well controlled then inspection of the transmitted intensity allows the position of a group of bubbles to be determined. The different methods used to control the light beam, and the choice made between using a spot of light or a line element, give rise to the many different machines that have been constructed.

There are two separate computing requirements for this measuring stage. Firstly the machines themselves need complex control and this is now normally provided by some small control computer connected on-line. Secondly pattern recognition procedures are required in order to separate the tracks of particles that belong to the event of interest from the noise present on the photograph, including all the other particle tracks that were photographed at the same time. In some systems all the tracks in the photograph are measured by the machine and then the pattern recognition is performed later. Another approach is to provide greater on-line control of the measuring machine so that only the tracks of interest are measured, probably together with a few other tracks that are not easily classifiable as noise.

The third stage ("geometric reconstruction" or "geometry") involves the combination of the two-dimensional descriptions obtained from the measuring stage on the different stereo-views into a three-dimensional description. This involves a heavy load of floating point calculation and data organization, typically requiring forty thousand words of main storage and two to three seconds of central processor time for each event on a CDC 6600 computer.

The fourth stage ("rescue") is needed because experience has shown that a significant proportion of events, typically ten to thirty per cent, will not have been well measured when they are first processed by a film analysis system. Various methods are used to overcome this problem, varying from a complete new pass through the system for all bad events to special procedures for "patching-up" the failures on interactive graphics systems.

In the classical approach to film analysis all of these stages have been kept separate. Partly this was because the importance of using the great potential of the geometric reconstruction stage to resolve difficult problems arising during pattern recognition had not been appreciated. It was also partly due to a feeling that the measuring device itself was the bottleneck in the system, and that above all its throughput had to be optimised. We now have a better understanding of the economics of film analysis and we realise that the book-keeping required to keep account of the progress of events through many separated stages is expensive, and that the throughput of the measuring device is not as vitally important as was once believed.

3. ERASME DESIGN PHILOSOPHY

In 1970 CERN started to take decisions concerning the provision of a film analysis system for film due to come in 1972/73 from a new bubble chamber called BEBC (Big European Bubble Chamber). For various technical reasons the possibility of modifying any of CERN's existing analysis systems was excluded and a decision taken to construct a new system. Various groups, particularly those working on POLLY[2] at Argonne and PEPR [3] at Oxford, had shown that since early hopes of completely automatic measuring systems with no operator intervention at any stage had proved unattainable, a better approach was to try to properly integrate the operator into the system.

A logical extension of these ideas led to a desire to carry out all the phases of scanning, measuring, geometry and rescue in an entirely on-line manner with the possibility of an operator dialogue being always present. The classic argument against this approach says that if you have a computer powerful enough to perform the geometric reconstruction with reasonable response time then you will waste an enormous amount of money because for most of the time this powerful computer will sit idle, while the operator is scanning the film or else is thinking about what rescue procedures are required. The ERASME design (4) (5) solves this problem by allowing several operators simultaneous access to this computer. Eventually five units, for scanning, measuring and rescue (which we will call S/M units for short) will be

attached to the central computer of the ERASME project, a PDP-10.

Each S/M unit consists of a number of elements designed to help the operator to scan, measure and rescue the events of interest in an efficient manner. In terms of CERN development effort the most important element is one based on a high precision cathode ray tube designed for computer-controlled measurement of the events. It also includes equipment for the transportation of the four views of BEBC film with a mechanism to bring any view into either an optical projection channel or a channel for the measuring equipment. In addition the operator has access to two digital displays; a track ball that enables him to point in either the optical or the digital display system; function buttons; an alphanumeric keyboard and a foot-pedal. To provide good modularity each S/M unit has all its elements attached to a control computer, which is a PDP-11.

The connection between the PDP-11's and the central machine is by means of a link constructed by DEC's Computer Special Systems group to CERN specifications. This link maps free UNIBUS addresses of our PDP-11's into PDP-10 main storage, packing two 16-bit PDP-11 words into each 36-bit word. The mapping allows the central machine to control the location of both a read-only and a read-write area for each PDP-11. The read-only area can be used to hold pure code, and can be shared among several PDP-11's. The read-write area is primarily used for the inter-machine communication. A more complete description has been published elsewhere (6).

One of the major features of the ERASME system is the long time-scale of the project. The fifth S/M unit, for example, is unlikely to come into full use before 1975 while geometry results on film measured on the first S/M unit were first obtained in June 1972. This is because of the constraints on the rate of production of equipment constructed 'in-house'. Another point is that BEBC is radically different from any other bubble chamber existing in Europe and we would like to have some experience with measuring BEBC film on the first S/M units before finalising the design of the later ones.

It is, therefore, clear that the ERASME system must function for a considerable length of time with a mixture of S/M units in both production and development status. In terms of hardware development the modular design is very useful, since much of the equipment can be set up and tested without needing any access to the central computer. The requirement for overlapped production and development is likely to continue for even longer for the software. This is because of two factors. The first is that starting to process a new experiment on an analysis system invariably requires some changes in the programs that control the film measurement and geometric reconstruction. These changes may be to adapt the programs to the particular topological configuration of the events of interest or else may be concerned with analysing film from a bubble chamber not previously processed. The second factor is that we will wish to study different techniques of measuring, etc. For example we may try to use one S/M unit in a much more automatic manner than the standard ones.

One method of overlapping production and development is to schedule separate periods of the day (and night !) when each activity can take place separately. We felt that this solution was unacceptable. Therefore we allow the two activities to proceed simultaneously although it is recognised that the production exploitation of the S/M units will be at a reduced level of efficiency while development is taking place.

This fact, that production and development must be overlapped for several years has a dominant influence on the specifications for the central machine software.

4. SOFTWARE REQUIREMENTS FOR THE ERASME PROJECT

(1) The Operating System (which we will normally abbreviate to OS) must be prepared to deal with a considerable number of concurrent jobs. The real point here is that the requirements of overlapped production and development exclude the possibility of using one monolithic program to control the procedures of scanning, measurement, geometry and rescue on all of the S/M units. Such a program would prefer that all S/M units worked in an identical manner. Since that is not possible it would like the flow of program control to be controlled by data that was particular to each S/M unit. However all possible program flows then need to be incorporated into the program when it is initialised at the start of each period of operation. That clearly is not flexible enough for the situation where you are trying to bring a new S/M unit up to operational status, or investigate new measurement techniques. Any attempt to put real flexibility in the hands of such a monolithic program requires writing many resource allocation and scheduling algorithms that logically should be part of the OS.

As a minimum therefore there will always be one job active per S/M unit and in addition several jobs might be expected to be simultaneously active to support the software development effort.

(2) It is clear that though there will be differences in the programming required for each S/M unit, much of the code will be identical for several or even for all S/M units. In the interests of minimising storage occupation this identical code should not need to be duplicated.

(3) When an S/M unit under development is

started up in parallel to production use of the other S/M units there will be an increase in the overall requirement for main storage. In order that the finite size of this main storage does not exclude such development activity there should be facilities to swap programs, or parts of programs, to and from backing storage.

(4) It should be easy to suspend a program that makes a request for activity at an S/M unit and to resume the program when the activity is complete.

(5) Communication between programs and the S/M units should be flexible. The amount of code that has to be incorporated into the OS to provide for this communication should be minimal and it should be possible to modify the communication specifications, for example by introducing control of a new device attached to an S/M unit, without regenerating the OS.

(6) It should be possible to 'tune' the scheduling of the overall system. The ability to give high priority to certain activities would be useful and it may also become desirable to give an especially low or high priority to all activities carried out by specified S/M units. For example, in order to improve operator response all production S/M units might be given a higher priority than development S/M units, and an S/M unit operating without an operator might be given an especially low priority.

(7) Complete program development facilities must be continually available. It should, for example, be possible (perhaps at some cost to the response of the production S/M units) to compile, load and debug fairly large FORTRAN programs at any time.

5. CURRENT IMPLEMENTATION

We first give some idea of the project's status. The central computer was delivered to CERN in June 1971, and a special interface to connect two control computers to the central computer was delivered to CERN in January 1972. The first S/M unit was assembled in a form that is a good approximation to its final design in May 1972. By June 1972 we had measured and reconstructed our first event. At the time of writing (October 1972) we are in a semi-production state on the first S/M unit. By March 1973 the first S/M unit should be capable of full production and the second unit should be nearing completion. Accordingly we have software running that controls a single unit system and we have firm designs for the control of a two unit system. In this Section we describe how our current software operates, and in the next Section we outline our plans for future development.

The manufacturer provides a time-sharing OS for the central computer that allows simultaneous access to the machine for many users via teletypes or teletype-like devices. This OS supports many simultaneous jobs (ours is configured for 20); it provides for job swapping to and from backing storage and gives users continual access to good program development facilities. We run without any changes to this OS, in particular there are no modifications to control the link to the S/M units.

The current system requires the use of three programs; one job to control the link, one job that processes all the operator interaction during the scanning and rescue stages and also provides control for the measurement stage, and finally a job to make the geometric reconstruction.

The job that controls the link to the S/M units makes extensive use of some real time facilities provided by means of OS calls. One such call allows a privileged user program to specify that a special device, such as the link, is its property and that when an interrupt is received from that device the program should be given control with relocation/protection in operation. Among other requirements a job making such a request must not of course, be swappable since the interrupt handling code cannot be on backing storage at the time of interruption; however an OS call to 'lock' such a job in main storage is provided. The actual communication between central computer and control computer is by means of interrupts and an area of main storage to which they both have read/write access. When a program needs something done at an S/M unit it writes some detailed specification of the task required into this communication area and interrupts the control computer at the specific interrupt vector related to the task. When the control computer has finished each interrupt driven task it writes the output data back into the communication area and interrupts the central computer in its turn. While the control computer is executing the task, the program that made the request will probably have made an OS call requesting suspension. The interrupt handling code in the central computer will normally make a call to request resumption of this suspended job.

The fact that the job that processes the interaction with the S/M unit operator does not have direct control of the link is mainly due to the requirement that jobs receiving interrupts must be 'located' in main storage, and we therefore wish to minimise the size of such jobs. Programs in the central computer can have two segments, with the virtual addressing being contiguous inside each segment but not between the end of the 'low' segment and the start of the 'high' segment. There are OS calls to enable two programs to share a common high segment and also to turn off the write-protection of the high segment, which normally holds pure code. We use this mechanism to communicate between the job controlling the link, which is locked in main storage and the job that handles the operator interaction, which has a swappable low segment.

In fact communication between the operator interaction job and the geometry program is handed in the same way. When the operator interaction job has completed its first measurement of an event it makes an OS call to start the geometry program and then suspends itself. The geometry program attaches to the high segment if that is not already in its virtual address space and then proceeds to execute. When complete it writes details of what it has discovered, including any requests for the operator to take rescue action into the high segment and starts the operator program.

6. FUTURE SOFTWARE DEVELOPMENTS

A comparison of the current implementation described above with the software requirements outlined in Section 4 shows that we have not discussed very much the questions of code sharing in a multi-unit system and of tuning the overall scheduling. The question of the scheduling can really only be discussed usefully after the multi-unit system has been implemented. The OS calls that allow privileged jobs to enter high priority queues and thus pre-empt jobs in queues of lower priority for the use of system resources allow us considerable flexibility for such tuning.

The topic of code-sharing in a multi-unit system is of more basic importance. The three programs that control the single S/M unit in the present system require about seventy thousand words (70K) of main storage, of which about 10K is data structure describing the state of measurement on the S/M unit. We clearly cannot run a multi-unit system by duplicating these programs for each unit.

As mentioned previously, because of the continuing development requirements of the project, the program versions needed for each S/M unit will be very similar but not identical. We intend to subdivide the programs into small tasks so that an S/M unit needing special treatment would load its private copy of certain tasks while using the common copy of all other tasks. Such a scheme requires that the common tasks are re-entrant for use by many S/M units.

It is probable that we will upgrade during 1973 our existing KA10 central processor to a KI10 central processor. The hardware of this processor permits multi-segment rather than just two-segment jobs. Thus we will obtain extra freedom in two areas. Firstly it would be reasonable to map the tasks discussed above into segments and then it becomes possible to swap infrequently used tasks on an individual basis. Secondly part of an S/M unit's data structure to which the control computer should have continual access could be 'locked' in main storage without obliging the remainder of the data structure to be similarily 'locked'.

We hope to report on our final organisation at a later date.

7. PROBLEMS OF THE APPROACH

We would ask four questions of any software project :

- Does it work ?
- How much effort did it take ?
- Is it efficient in terms of size and speed ?
- Where can I find details ?

As we indicated in the previous section we have reached a semi-production status on our first S/M unit about three months after this unit first approached a reasonably final state. During periods when the unit is being used to measure film other users can still use the timesharing facilities of the central computer. Our approach has therefore worked up to the present. We have no indications that it will not be satisfactory as further S/M units are added to the system.

To date the applications-oriented software effort on the central computer has included project design, adaption of the standard CERN geometry program for our needs, provision of a new program to control the CRT-based film measurement system and handle all interaction with the operator, and provision of test and where appropriate calibration programs for all aspects of the S/M unit. The effort involved is about eight man years. We divide the systems-oriented software effort into two parts. One part includes jobs, such as acceptance testing of the main computer, that would have been necessary regardless of the approach adopted and this has required about three man-years work. The other part includes gaining familiarity with the manufacturer-supplied software and providing special system facilities, such as the link control job, a job to load and dump the PDP-11 over the link, and the general organisation of the single S/M unit software. This has required about two man-years of effort and it is this effort which must be contrasted with the effort that would have been needed had we decided to provide a purpose-built OS.

Our approach is certainly not the most efficient possible in terms of size. Any manufacturer must provide an OS of reasonable generality and it is clear that a purpose-built OS should be smaller. Our current OS is configured for 20 jobs and occupies 32K words of main storage, with no overlaying of code. Of this 32K we might save 8K fairly easily since we could afford to provide user programs to handle most peripheral devices rather than having the interrupt handling code permanently resident, and we certainly do not use much of the code provided in order to optimise disk utilisation in systems with many disks and controllers. It does not seem likely to us that an OS that would be of sufficient generality to support the ERASME software would occupy less than about 24K of main storage. If we find this space inefficiency to be a problem at a later time then we believe that we can save space along

the lines outlined above without making enormous efforts.

The argument that a purpose-built system should be smaller than a general one should perhaps be extendable to say that a purpose-built system should also be faster. We believe that this is much less likely to be true. The speed of response to OS calls, which is our major concern, is determined by three factors, the overhead required to protect other users, the generality of the OS call definition and the efficiency with which the code was written. In a purpose-built system it might be possible to take risks and provide less inter-job protection but otherwise there does not seem to be any scope for significant improvement in our experience. It is also unlikely that special-purpose scheduling would have been useful for us, since it would have been too inflexible. We remark that in general users of a timesharing system obtain a much more personal feeling for machine efficiency than users of a batch facility.

Finally we believe that our approach has helped us in the area of documentation. Besides not having to write and debug a large OS we do not have to document it either. The manufacturer does this for us fairly efficiently, although we have to pay for the service. A similar argument says that we are using software that is used by many other installations and the fact that the maintenance and correction of faults in this software is carried out by the manufacturer and not by us is largely to our advantage.

8. CONCLUSIONS

We believe that if we had adopted the approach of trying to create a tailor-made operating system for our project then it would have been necessary to :

- Separate production measurement periods using this tailor-made system from software development periods using a normal operating system.

- Devote significant amounts of software effort to provide substantially the same features as are already present in the PDP-10 operating system for handling many simultaneous users, job swapping, code sharing, passing interrupts to user programs, suspending and resuming jobs, and scheduling on the basis of priority queues.

Such a tailor-made system might be smaller than using the standard product, and the time overhead for certain activities might have been less. We believe however that our current approach has allowed us to reach production status more quickly and with less effort than would have been the case if we had written a tailor-made system. When operating system overheads are seen to be of significant importance then effort can be devoted to reducing them.

9. ACKNOWLEDGEMENTS

We are grateful to D.H. Lord and E. Quercigh who are the leaders of the ERASME project, for their support and to many other members of the ERASME group for their assistance and ideas.

10. REFERENCES

(1) P. Zanella, "Machine Recognition of Patterns in Particle Physics", Proc. of the First European Conference on Computational Physics. April 1972 (in print).

(2) W.W.M. Allison et al., "Automatic scanning and measurement of bubble chamber film on POLLY II", Proc. of the Int. Conference on Data Handling Systems in High-Energy Physics, Cambridge (U.K.), 1970. CERN 70-21, pp. 325-356.

(3) J.P. Berge et al., "Oxford PEPR system", Proc. of the Int. Conference on Data Handling Systems in High-Energy Physics, Cambridge (U.K.), 1970. CERN 70-21, pp. 61-88.

(4) D. Lord and E. Quercigh, "The ERASME Project Summary". CERN-DD/DH/70/21, September 1970.

(5) The ERASME Group, "The ERASME System, Concepts and Current Status". CERN-D.Ph.II/ERASME 71-35, November 1971.

(6) J. Bettels et al., "A High Performance PDP-10/PDP-11 Link", Proc. of the 8th DECUS Europe Seminar. September 1972 (in print).