EE

EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN - SL DIVISION

CERN SL/94-66 (DI)

SW 9434

Requirements and Trends in the Control of Accelerators

J. Poole

Abstract

The primary requirement for any accelerator control system must be to enable the operator to deliver the accelerated particles to the users in an effective and efficient way. This breaks down into separate requirements for the equipment, the control system hardware and for the software. The requirements and the extent to which control systems satisfy them are examined. Whilst there is no single direction in which control systems are converging, there are a number of trends reflected in the main thrust of new developments. These trends and their consequences are discussed.

Fourth European Particle Accelerator Conference (EPAC 94), London, United Kingdom, 27.6-1.7.94

Geneva, Switzerland
5 August, 1994

Requirements and Trends in the Control of Accelerators

John Poole
SL Division
CERN
CH-1211 Geneva 23

Abstract

The primary requirement for any accelerator control system must be to enable the operator to deliver the accelerated particles to the users in an effective and efficient way. This breaks down into separate requirements for the equipment, the control system hardware and for the software. The requirements and the extent to which control systems satisfy them are examined. Whilst there is no single direction in which control systems are converging, there are a number of trends reflected in the main thrust of new developments. These trends and their consequences are discussed.

1 INTRODUCTION

In the construction of an accelerator the control system presents a wider range of problems than usual because it has to unite all other systems. *The control system* means different things to different people depending on whether they are software, hardware, or equipment specialists or users and in this paper I express my views primarily as a user.

To a certain extent the control system hardware (which physically connects the equipment to the operator) has evolved to a stage where it can be built from standard parts. The application software, which represents at least half of the total software effort, is not as well understood as that which runs close to the equipment. The emphasis in this paper will therefore be on the application software although there is some discussion of hardware issues.

An accelerator is built up from a wide variety of hardware and software components which have to work in concert to deliver the particle beams to the user. Such complex assemblies are necessarily composed of high technology systems built by specialists whose expertise rarely covers more than a small part of the range involved in the whole project. It is difficult to define the overall requirements for the accelerator because the operators, builders and users are usually people from different disciplines. One way to overcome this difficulty is to analyse the needs of the people who have to use the control system to produce the beams and then match the control system performance to these requirements.

Trends in accelerator controls reflect the increasing sophistication of the accelerators and the need to optimise the use of limited manpower. Full use is made of the computing power available in today's market and more and more hardware (analog) functions are being replaced by software processes. This extends to the use of fast feedback loops which enable one to compensate for the limits of the accelerator hardware. At the same time, there is a wealth of commercial software which is being incorporated and sharing of hardware and software between laboratories. The level at which control is performed continues to get higher: in early machines one would control parameters at the level of magnet currents by physically changing potentiometers on the power supply, today it is common to control abstract quantities like the imaginary part of the coupling matrix.

2 REQUIREMENTS

Accelerators are built to deliver beams to the users and the control system is there to allow someone to operate the accelerator so that the users' requirements are met in an efficient way. Although this is the primary requirement it is often not perceived as such before, or during, the construction phase because the people building the system are only aware of the local constraints on their part of the system. The main user of the control system (the operator) has to deal with the whole machine, not individual components and if the necessary coherence and flexibility does not exist, operation at best will be inefficient. This problem is probably addressed better in the smaller labs where fewer people are concerned with the accelerator construction and operation.

In the accelerator field we have always been very good at building high performance, sophisticated hardware but there are recent examples of accelerators which have very high performance control systems (in terms of remote control) but which are extremely difficult to operate. This kind of situation clearly indicates that some fundamental changes in our methodology are required.

The primary requirement can be analysed into components which form a kind of pyramid below it. For the operator to satisfy the users' needs he must be able to deliver the required intensities, energies etc. and this in turn means that he has to be able to set, measure and correct the machine parameters. Continuing with this analysis leads through to the lowest level, where information is exchanged with the hardware. At the same time the secondary requirements coming from the equipment specialists and accelerator physicists can be folded in.

In the following sections I will highlight some of the requirements common to many accelerators which can influence the way their controls are built.

2.1 *Control System Hardware*

The control system hardware should provide the platform(s) for running the control software, timing, synchronisation and the communication between the software and the equipment. The specification depends on the performance requirements like scale, speed, need for feedback loops, global control of distributed equipment, timing, etc. and can limit the accelerator performance. When these requirements have been elaborated in some detail, one can start an iterative procedure in which the hardware configuration and the functional requirements are matched.

2.2 *Modeling*

The modeling program is the starting point for the design of an accelerator and is therefore the source of many of the design parameters. During the life-cycle the model will remain at the heart of development and therefore plays an important part in control because it will continue to be used to describe the physical layout and to produce settings. Integrating (or at least interfacing) the model program with the control system at a very early stage provides a single source for the accelerator definition and a point of unification for all groups in the project.

2.3 *Data Management*

The data in a control system represents our knowledge of the accelerator and it is essential to be able to correlate and analyse it. A coherent solution is required if we are to reach many of our goals (save-and-restore, reproducibility, understanding of the physics etc.). Such a system can only be achieved by carefully analysing the information structure and identifying the various elements and their relationships. From the analysis it is possible to create a design for the data structures which can then be implemented in the data management system.

Data management is fundamental to any control system and a well chosen system will simplify many applications and give a coherence to the software. It is essential to establish the data management design before starting the design and implementation of the application software. In the past a lot of effort has been spent on building in-house data management systems which were costly to maintain and which did not necessarily adapt well to hardware developments.

2.4 *High Level Applications*

The operator uses high level applications to translate physics requirements into beam behaviour. These applications run in an environment comprising the man-machine-interface (MMI), data management system, graphics packages, timing/synchronisation system, sequencer etc.
Some typical examples of high level applications are:

- Measurement and correction of beam parameters
- Control of the evolution of hardware parameters in time
- Read and set the hardware
- Save-and-restore
- Alarm system
- Data analysis/display, correlations ...
- Personnel safety and access

In many accelerators there are high level applications which contain complex algorithms combining more than one of the applications listed above. This kind of application embodies the understanding of the complex mechanisms in the accelerator and allows the performance limits to be raised in operation. The specialist knowledge is built into the application and the operator is able to tune the performance easily because the complexities are dealt with by the application.

2.5 *Hardware De-bugging and Maintenance*

There is a requirement for hardware experts to be able to de-bug and maintain their equipment and the software for this requires less abstraction but more functionality. It is very useful for a hardware specialist to be able to inspect faults from the same environment as the high level applications i.e. he can sit with the operators in the control room and investigate problems but he will also need access close to the equipment. His requirements for the software are very different and it is therefore not reasonable to expect him only to use the high level applications. In general the hardware specialist's software should use the same underlying packages to access the equipment but it is not cost effective to wrap his software in a highly polished MMI.

2.6 *Driving the Hardware*

There is a fairly restricted set of operations that one can perform on hardware (RESET, ON/OFF, SET, READ, ENABLE, LOAD FUNCTIONS ...) and most equipment uses a fairly large subset of them. This means that the software modules which drive the hardware can be identical as long as there are hardware-specific modules which take care of the particularities of the equipment in question. Using this technique means that, seen from a higher level in the system, a collimator will look much the same as a power supply. By separating this reduced number of driver modules from the algorithmic parts, one can avoid duplicating functionality and save effort.

2.7 *Beam Parameter Control*

Although the requirements for measuring many beam parameters are very simple, the great diversity of instruments causes problems. Further difficulties arise from measurements which require the co-ordination of settings changes and correlation of data from several instruments. It is always important to record the environment in which a measurement was made, so correlation of measurements across the whole accelerator is essential. This latter requirement is hard to fulfill if the system is not considered in its totality.

The algorithms at the heart of many of these applications can often be very general and hence applicable to many different accelerators. This code, though sometimes lengthy to develop, only represents a small part of the application because a lot of effort is spent wrapping the algorithm with the MMI and connecting the output to hardware. A suitable design of the application software would allow one to easily 'plug in' new algorithms. The possibility of sharing algorithmic parts of applications is interesting but does not yield large savings in effort.

2.8 Displays

A feature of any control room is the display of machine and beam parameters. There are usually a handful of parameters which the operator watches as he makes tuning adjustments but optimisation becomes virtually impossible if the display (or measurement) of the relevant parameters is slower than the adjustments. The operational performance of the accelerator is therefore directly related to the effectiveness of these measurement/display systems. Their requirements will directly determine several performance criteria for the control system and instrumentation and should therefore be given appropriate emphasis.

2.9 Archival and Logging

The understanding of an accelerator's behaviour often requires analysis and correlation of unexpected parameters; one therefore needs to log as many parameters as possible and have tools for analysing the data. The cost of disk storage is low enough that one is no longer concerned about keeping large volumes of data but it is obviously worth taking some data off-line (archiving) for maintenance reasons. There are many instances where analysis of logged data has led to new insights into the behaviour of machines and this certainly justifies putting some effort into building these systems.

Save-and-restore is a closely related topic which should dovetail with the general archival and logging system; it is fundamental to the operation of any accelerator but it can be very difficult to achieve. The requirement is to be able to reproduce the accelerator's performance from some previous time which means saving full cycles (how one got there) as well as snapshots (which can only give one value for the settings).

The key factor in this area is being able to uniquely identify and save all of the relevant measurements, environmental factors and setting parameters which determine the performance at a given moment. In addition one has to be able to retrieve the saved data in an efficient way – looking through a log book for a filename is not the best way.

2.10 Alarm System

An effective alarm system provides equipment protection and increases operational efficiency. The system should monitor the equipment, record and analyse faults and report warnings and faults to the operators. The system should also be capable of informing the operator what to do when there are problems with the equipment. This system is somewhat independent of the mainstream of operation but is conditioned by the operating mode and is concerned with the same equipment. The construction of such a system is therefore bound to the other parts of the system.

3 TRENDS

There is almost a religious fanaticism about many facets of control systems which is unfortunate: this may not be a particularly new trend but it is certainly one which influences development. It is one reason why there is not one particular direction in which control system development is heading; there are new developments in both PC and UNIX -based systems, C and FORTRAN are both used and so on. The reason for the diversity is simple; there is no single solution to the problem, in fact it is something of an achievement to have so few directions.

The increasing cost of building control systems is driving development towards the use of commercial systems and sharing of systems between laboratories. A typical example of this is the development of EPICS[1] and its proliferation around the world.

Although the majority of recent machines have been light sources which have, for the most part, adopted control system philosophies from other labs, the new developments have tended to come from the larger projects.

3.1 Control System Architecture

An interesting trend is the increasing acceptance of a standard configuration featuring a layer where the application software runs which is connected through a TCPIP network to a layer where the software which deals with the equipment runs. Such a trend opens the way for much more sharing of software modules and the creation of more generic systems.

3.2 Commercial Systems

Commercial control systems are being introduced at many facilities around the world and this is clearly a trend which will continue. Such systems are particularly suited to the control of industrial systems which are part of the accelerator (cryogenics plant, cooling plant etc.). It is important to integrate the systems because they are in the accelerator environment and may affect performance e.g. through the temperature of the magnets.

Another field in which commercial software is playing an important rôle is in database management systems. As our accelerators have become larger or more complex there are increasing volumes of data to manage; commercial systems offer a way to handle this problem and they also provide a standard interface to the data. It is now possible to handle Gbyte databases in an efficient way and even use databases for on-line control and these possibilities are strongly influencing the development of new systems.

Graphics packages (in the widest sense of the term) were perhaps the first area of control functionality to be served by commercial software. Their use is still common today and even though many are based on standards like MOTIF they are not often portable across platforms.

An unfortunate consequence of using commercial systems is the frequent need to upgrade. Manufacturers are continuously improving the performance of their products and producing new versions which means that old versions are no longer supported. In this situation one can find that an addition or upgrade in one part of the system can have an enormous knock-on effect (e.g. addition of an optical disk which requires a later model workstation ⇒ an operating system upgrade ⇒ move to the next version of the RDBMS ⇒ re-make all the application software ...).

3.3 Feedback Loops

In recent years a number of feedback systems have been built which have extended the performance of accelerators. In many cases these have been introduced to compensate for some hardware or environmental defect i.e. they have been added as a fix. An interesting development is the incorporation of such loops in the equipment design in order to reach higher levels of performance.

3.4 Software Sharing

An effort is being made to share software at the lower end of the control system[3]. This interesting trend has certainly been facilitated by the move towards a standard architecture.

Perhaps a more significant development is the interest in sharing software at the application level. It has been accepted that the basic functionality for many accelerators is the same and therefore one ought to be able to share applications. Algorithms have always been shared among laboratories: MIKADO, the orbit correction algorithm, was originally developed at CERN and many years later it reappeared in the PS Division via Brookhaven Lab.

A new concept emerging from this discussion is that of a software bus: this would enable one to glue various applications together and even manipulate the data flowing between them. There is now intense activity to establish an international standard in this field, reflecting the interest across the board in this kind of software development.

3.5 Methodologies

In recent years the commercial world has woken to the need for methodologies in software engineering and the accelerator world is following. Many of the recently built control systems have used formal methods like Yourdon/DeMarco during analysis, design, implementation and commissioning phases. These techniques are extending beyond the confines of application software and into information analysis and database design. The adoption of such formal methods opens the possibility of sharing analyses as well as designs. Because much of the functionality of accelerator control systems is common, sharing of the concepts described in the functional analysis becomes a useful exercise. It is clear that this sharing of concepts has always been used, but previously the concepts were not expressed formally and the sharing relied on meeting the developers and discussing the ideas. Within the work of the EPCS Group on Software Sharing[2, 3] some progress is being made in this direction.

A revolution in the software development process is beginning in accelerator laboratories and this will certainly take hold over the next few years. The acceptance of formal methods and use of ESA standards are examples of this change but there remains much to be learnt about the management of the software development process.

Object oriented methods have been a feature of accelerator control systems for many years, the difference today is that standard implementation methods are becoming available. Object technology will probably reach maturity in time for full exploitation in the next generation of accelerators and should make re-use and sharing of software easier.

3.6 Software for Commissioning

A second version of the application software often appears at a very early stage in the life-cycle of an accelerator. The first version usually covers the basic functionality but it is built before the relative importance for operation of the various processes is known. Once the machine has been commissioned it is possible to see where the problem areas are and how one can improve performance by shifting the emphasis in the application software: new/better displays, logging of parameters, new algorithms and so on.

By recognising that such changes are inevitable it is possible to save resources and implement a minimum set of software which can be used for the commissioning and early stages of operation. If, for example, it is found that orbit correction dominates operation then the orbit software should be highly tuned and perhaps even a feedback loop should be built. These changes do not represent a change in the underlying functionality, rather some additional performance requirements.

4 CONCLUSIONS

The traditional method of building a control system from the bottom up continues to be common practice. I hope that I have demonstrated in the above that this causes many problems because one cannot see how the physics requirements can be satisfied by the ensemble of equipment. This is not a plea for a strict top-down approach to the problem, more pointing out that one must try at an early stage to understand what one is trying to do and to match the performance capabilities of the control system to the requirements.

The attitude that one can only build a system using a certain type of hardware (e.g. UNIX workstations/VME) is clearly not valid. There are instances where this might be the case but it should not be the starting point in building a control system. It is a sad observation that people are still building control systems which do not, and in some cases cannot, satisfy the requirements.

The high level application software plays an important rôle in the machine performance and should therefore receive appropriate emphasis. The key to success is a good understanding of the requirements and building a coherent and extensible system which satisfies them. The requirements for many aspects of application software for an accelerator are well understood and can be shared from one machine to another.

The control loop between operator and beam is closed through the displays of machine/beam parameters. Careful attention to the choice of what to display and the performance of the measurement/display system yields dividends in accelerator performance.

The increasing use of formal methods is helping to avoid some of the problems in the construction of control systems. It is helping in the trend towards greater sharing of control systems, which is good news. With increasing awareness of the problems across the accelerator laboratories there is hope that future control systems will be built more efficiently and will satisfy the requirements better than in the past.

5 ACKNOWLEDGEMENT

I would like to thank my colleagues in accelerator laboratories in Europe and the USA for many useful and stimulating discussions.

6 REFERENCES

[1] M. Knott, et.al, EPICS: A Control System Codevelopment Success Story, in Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'93), Berlin, Germany, October 1993.

[2] Interdivisional Group of the European Physical Society (EPS) on Experimental Physics Control Systems.

[3] B. Kuiper, Joint Statement from the One-day Workshop on Software Sharing, in Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'93), Berlin, Germany, October 1993.