MODERN ACCELERATOR CONTROL SYSTEMS

K. Furukawa*
High Energy Accelerator Research Organization (KEK)
Tsukuba, Ibaraki, 305-0801, Japan

Abstract

Modern approaches to accelerator control systems are discussed on the basis of several criteria including software and hardware implications while maintaining reliability under changing requirements.

INTRODUCTION

Recently developed particle accelerators are significantly advanced and they require well-arranged operational tools. One of the major components of such a particle accelerator is the control system. We have performed several trials in order to develop control systems for past and present accelerator projects. Furthermore we have obtained practical solutions and are evaluating possible additions to these systems.

Firstly KEKB and Linac control systems are described as examples of accelerator control systems and operational environments. Then, accelerator controls are discussed in general from a practical viewpoint. Typical technologies available for controls are described in the next section. Finally, a discussion on reliability is also presented for practical applications.

KEKB AND LINAC CONTROLS

There have been several control systems in KEK. KEKB and Linac control systems are briefly described as typical examples of control systems.

KEKB Control System

KEKB is an asymmetric collider B-factory for the study of CP-violation with 8-GeV electron and 3.5-GeV positron rings. In order to achieve high luminosity and obtain good experimental results many active parameters are tuned and improvements are made daily and, as a result, requirements to the control system have been changed.

The KEKB control system is a standard EPICS system (experimental physics and industrial control system) with approximately 100 VME-based IOCs (I/O controllers) [1]. For the field interfaces, 200 VXI mainframes through MXI interfaces, 50 CAMAC crates through serial highways, and 200 ARCNet segments are installed in addition to the VME frames. Many GPIBs, RS232C devices, and PLCs are also employed. Control services are provided by different types of Unix computers such as HP-UIX, Tru64 Unix, Solaris, Linux, and MacOSX. MacOSX computers manage most of the data processing tasks and graphic displays.

* < kazuro.furukawa @ kek.jp >

Because of a five-year gap between the previous project TRISTAN and the KEKB project, majority of the software and hardware resources was reconstructed, while some CAMAC resources were reused.

Software for EPICS R3.13 and R3.14 applications is developed with Capfast, MEDM(DM2k), etc. Approximately 250,000 EPICS records are spread over the 100 IOCs. While some of the hardware-related algorithms are programmed in dedicated record types or as record links, most of the control and operational algorithms are implemented using scripting languages such as SADscript/Tk and Python/Tk. Scripting languages are used because they are developed rapidly and they can be used by a variety of users such as physicists and operators. The disadvantage of using IOCs for programming algorithms is that they have to be rebooted when new records or record links are implemented.

Among more than 200 operation programs, the KEKBlog archiver and viewer, KEKB-alarm alarm handler, Zlog operational logbook system, and KEKB optics panels are some of the most frequently accessed operation tools. The KEKBlog archiver stores data more than 2 GB of data every day.

Linac Control System

The electron/positron Linac has been in operation since 1982; it was upgraded for KEKB injection in the period of 1994 through 1997 with 8-GeV electrons and 3.5-GeV positrons. Its length is 600 m, and it has 60 high-power rf stations and 400 magnets. Since it was in operation for PF injection during the upgrade, it utilized many components from the previous projects.

Linac provides beams with different characteristics to KEKB, PF and PF-AR rings, and the beams are switched more than 300 times a day. Since these rings are factory machines, the upstream Linac is required to carry out a reliable and stable beam operation and to precisely control the beam characteristics such as Twiss parameters, timing, and charge. Furthermore, new operational beam modes are added almost every year [2].

The Linac control system was revived between 1991 and 1993 just before the approval of KEKB, and minor and gradual modifications were made during its upgrade for use in KEKB. The control system comprises 30 VMEs, 150 PLCs, 15 CAMACs, 30 VXIs, many Unix computers, and redundant Ethernet/IP networks. 20 oscilloscopes with built-in 3-GHz computers will be introduced soon.

The design concept of this system was based on the use of de-facto standards such as Unix, VME and TCP/IP, and

the use of optical Ethernet/IP networks for all device controllers without any special field networks. This concept was inherited by J-PARC controls while the use of EPICS was inherited from KEKB controls [3].

Most of the communication in the control system is achieved by locally developed RPC (remote procedure call). The overall system is multitiered; the lower level is controlled by UDP-RPC or simple UDP protocols in order to recover failures promptly. The upper level is controlled by TCP-RPC, and a network-wide shared memory system is provided for read-only information. The Linac API (application program interface) provides transparent access to UDP-RPC, TCP-RPC and shared memory.

This system provides three views of the Linac accelerator and beam, namely, an engineering view for devices, an operational view of the overall accelerator, and a scientific view for beam improvement. It also has communication links and gateways to console systems, utility facilities, and downstream accelerators including EPICS gateways.

EPICS gateways are implemented in several methods such as soft-IOCs, portable CASs (channel access servers), and dedicated IOCs with gateway programs. Presently EPICS gateways are utilized for most of the data archiving with EPICS channel archiver and KEKBlog, and for operational alarms with KEKB-alarm.

Operation of KEKB and Linac

The beam operation of KEKB and Linac is carried out by KCG (KEKB commissioning group), which comprises 40 staff members from the KEKB and Linac groups. Most of the operational panels are developed with scripting languages, and those for beam operations are written with SADscript [4].

SADscript is a Mathematica-like language whose processor is written in Fortran and has built-in interfaces to EPICS channel access (asynchronous and synchronous), Tk X11 widgets, CanvasDraw, Plotting, KBFrame graphic libraries on top of Tk, numerical data processing such as fitting and FFT, inter-process communication, and SADcore, a full accelerator modeling engine including symplectic beam tracking and beam envelope capabilities [5].

SAD and SADscript are designed to carry out almost all tasks related to accelerator and beam operation. The Mathematica-like list-processing functions of SADscript enable the rapid development of online operational software. Many novel concepts have been tested using such rapid prototyping just after the proposal. Many of these theories have been found to be impractical; however, some of them have proved to be inevitable for improving the accelerator. Rapid development is very important because of limited time changing circumstances. Virtual accelerators are also built with SAD in order to understand the behavior of new operational parameters.

There are several methods to define the accelerator beam-line lattice in SAD. However, it does not read a standard input format. It is desirable to develop a new format that covers beam-line geometry and beam optics.

ACCELERATOR CONTROLS

The definition and purpose of control systems are not determined until the technical details of the accelerator are decided. They are often redefined during the operation. Thus, control systems require flexibility as well as robustness.

History

Initially, the NODAL interpreter language environment was successfully used at SPS/CERN. Then, because of the need for more general software tools, the ICALEPCS (international conference for accelerator and large experimental physics control systems) was organized in order to share common ideas [6]. In the meantime, NODAL was used at TRISTAN/KEK, and SLC/SLAC built an advanced control system with many micro-computers and a VMS computer in order to realize a linear collider.

Around that time, the standard model of control systems was said to be a combination of field-network, VME, Unix (or VMS) and X-Window. Linac/KEK followed such trends with an Ethernet-only field network and scripting languages. Moreover, more software sharing between the accelerator projects was expected; however, the discussion often fell into the definition of a class object that represents a whole accelerator, which can be never achieved. More general tools such as ncRPC/CERN, TACL/CEBAF, and ACNET/Fermilab were devised. One such tool, EPICS/LANL-ANL became popular maybe because of its simplicity and its adoption at SSC, and many institutes including KEK participated in the development.

The object-oriented approach, which was expected after RPC, appears to be the recent trend. In comparison with other approaches, this approach can receive more benefits from software engineering. Further, CORBA (common object request broker architecture)-based tools such as CICERO/CERN, TANGO/ESRF and CORBA+Java/CERN have been developed.

Balance

In designing accelerator control systems, we may have to strike a balance between many concepts. In most cases it is not possible to choose one concept, because the system has to interact with many different subsystems. We consider implementations that optimize the strengths of different ideas under different criteria.

Object- and Channel-oriented Technology

Object-oriented technology definitely aids software development and maintainability based on the benefits of software engineering, which have accumulated for a long time. The software should be extensible, and it would naturally have cleaner definitions. However, different people

may have different ideas about control objects. CORBA and object-oriented languages provide these features.

Channel-oriented technology is simple and flat; therefore the system becomes scalable and can be understood easily. Although this technology would not benefit from the advantages of software engineering directly, there is a way to overcome the disadvantage if some more software tools are developed. Further, several technologies can be combined to build application software layers. Since this technology is simple, it can have gateway links easily to other architectures.

Compiled and Interpretive Languages

Normally, two-level languages are preferred; a compiled language for established algorithms and an interpretive language for the rapid prototyping of new ideas. The NODAL language was successfully used at SPS/CERN; however, it was not used at LEP since it was considered that software in a compiled language should be more policy-driven and manageable. However, interpretive or scripting languages can efficiently handle lists and many of them are now object-oriented; so that those languages enable more physicists to attend the direct software development. Another level of management may solve the maintenance issues.

Aggressive and Conservative

A new and aggressive technology is attractive; however it can only be a fashion or a fad. We need to watch those new technologies and the assimilation of their essence makes the project more active. However, for deploying them completely, it is important to ensure their quality and compliance to a de-facto standard.

The life of an accelerator is often longer than that of the user facilities and the commercially available technologies. The operational knowledge base is mainly stored in the form of software and database in the control system. Even well-known beam stabilization algorithms require practical methodologies. They are rather valuable, and the control system, which contains such information, should be maintained carefully in order to increase its lifespan.

International and de-facto Standards

Since international organizations tend to pursue ideal solutions, some of them fail to become industrial standards. In such a case it is not advantageous to employ these solutions. It is difficult to obtain a de-facto standard, however, if found, such a standard can prove to be advantageous.

Products can be selected from those already available in the market, thereby saving manpower and avoiding proprietary development. Also, a standard at one time would be provided with continuity solutions for the next generation. At CERN some projects select products based on market shares. As a whole, choosing a de-facto standard is better for long lifespan of the system.

AVAILABLE TECHNOLOGIES

There are many efficient technologies that are currently employed or are under evaluation. In this section, we discuss the reasons for the use of these technologies.

PLC

PLCs (programmable logic controller) are utilized at relatively slow controls because their rule-based algorithms can be adopted effectively in simple controls. Modern PLCs provide IP network connectivity for the both controls and management functions such as program development and condition monitoring of the PLC itself. Some of them provide socket-based communication with a response time of one millisecond over the network. Because of these features, 150 PLCs are installed at Linac/KEK for magnet, vacuum and microwave controls.

PLCs enable isolated controller development and the outsourcing of the device and the controller. Local panels can be attached, and many local maintenance functions, which improve the reliability of the devices, can be implemented.

For software development, an international standard, IEC61131-3, defines five programming languages and their use, with emphasis on naming. The standard itself provides many good insights into the control systems. It has not been very popular in Japan; however, recently many vendors have made significant attempts to realize the IEC61131-3 development environments, with the XML representation of resources. Similar to EPICS, this standard should enable shared development at the international level.

Ethernet/IP-only Networks

As described previously, a policy for using Ethernet/IP for field networks was chosen more than ten years ago. The decision was partly based on the preference to save manpower and avoid proprietary development. The policy was enforced after the inclusion of the TCP/IP software stack into Windows95 by Microsoft, which promoted the wide acceptance of TCP/IP technology in the industry; later the policy was inherited by J-PARC controls.

The policy enabled simple network management with commercially available network components, routing methods, network booting, failure analysis tools, etc. It also resulted in cost reduction even with optical network components, which are inevitable in the Linac gallery with high-power modulators. The IP technology continuously makes gradual transitions, which result in a longer lifespan.

Recently, many measurement instruments have been developed, embedding fast computers, which enable embedded EPICS IOC or MATLAB software. Unfortunately security issues arise when Windows is employed as the operating system.

FPGA

FPGA (field programmable gate array) is another technology that is commonly used in recent times. Using this technology, a digital circuit board can be designed rapidly even with embedded software. This technology is flexible and robust; therefore it is a wonderful platform for local controllers. We have to be careful that its flexibility does not lead to a lazy design, which may cause circuit bugs. More and more gates, memory, and i/o pins will be available with faster clocks. Furthermore better software support is expected.

ATCA and MicroTCA

ATCA (advanced telecommunications computing architecture) is a next-generation networked computer standard that emphasizes on reliability. A board can be connected with several 2.5-Gbps interconnects including GbE, PCI-express and 10GbE. The standard defines possibly redundant shelf managers that manage the robustness of the system through the IPMI (intelligent platform management interface) over I2C connections.

Many facilities such as hot swap, redundant CPUs, switches, and fans can be utilized for improving reliability. While cost reduction is expected in the future, currently the ATCA is somewhat expensive. However, It is selected as the main platform of the future ILC.

MicroTCA was defined in 2006 based on AdvancedMC (ATCA mezzanine card). Although the initial standard was limited, it has incorporated many facilities of ATCA. MicroTCA may not replace VME immediately; however it can potentially be used in medium or small facilities since, unlike CompactPCI, it is not directly connected with PC industries, which advance too rapidly.

EPICS

EPICS has been successfully developed in international collaborations. It is a de-facto standard in this field. However, there still remains a list of features to be implemented, and some of them are being resolved.

Currently, a naming scheme and/or design of new records provide some object-oriented design supports, however, it is preferable to obtain more software-engineering support. For this purpose several different developments such as JavaIOC, CSS (control system studio), and data access layer are promoted around the world.

While user- and host-based security mechanisms are available, for larger installations, enhanced protection features such as dynamic controls of protection and access logging are preferable.

It is also desirable to obtain a dynamic configuration of runtime database. Static database had partly led to the intensive use of scripting languages on the client side. The implementation of the dynamic feature may be shared with the redundant IOC project.

Magnet Controls

Magnets are typical control device objects; however, certain factors should be considered for using magnet controls. Complications arise partly from non one-to-one correspondence between magnets and power supplies. The conversions of several units in engineering and physics applications, such as current, field, kick, and momentum, are sometimes complicated and are dependent on calibration methods. Moreover, synchronous timing operation is often required for ramp-up, tune-change, etc. Standardization is another difficulty depending on the methodology.

Timing and Event Controls

Since accelerators are valuable, more and more time-sharing operations may be required. Thus, tighter coupling between control and timing/event systems may be required. Recently, an event system designed for SLS and DIAMOND has been shared between several projects, and it could be a good reference [7].

RELIABILITY

The end user may require robust operation of the accelerator; however, the accelerator itself should be flexible to enable continuous improvement. Thus, we may have to compromise between practical or ideal solutions under restrictions of safety, manpower, time, etc. The conditions also change during operation. Here, we consider the adaptive and practical reliability.

Possible Correct Solutions

Correct solutions can be obtained by careful consideration. If we have well-defined software class objects for accelerator controls, the errors in the design, coding and usage of software can be reduced. We expect to achieve this in the future.

Well-arranged naming conventions reduce human errors. They also enable the development of more computer-aided tools. We have partly realized these conventions depending on the accelerators. Well-specified deployment procedures are employed at some accelerators, and some of them can be automated to reduce the errors. However, the real situation is not simple and there are many exceptions. Thus, practical solutions are required.

Surveillance for Everything

Because we have written numerous software codes, which unfortunately assume certain circumstances, they will eventually fail. We manage and excessive number of computers and many network devices that may have assigned the wrong parameters or may have been equipped with inadequate resources.

Thus, we have to find the most important feature of these installations and obtain the simplest tests for them. Routine tests should be automated. If an anomaly is found, an alarm is issued, an e-mail is sent to the author or manager, and an

error log is recorded. If it is not critical, the related software or hardware is restarted; otherwise, the event is reported to the human operator.

This is not ideal, but it is effective and practical under conditions of limited human resources.

Testing Frameworks

We often move operational environments for improved resource performance including computers and development software. However, the event may lead to malfunctions because of incomplete compatibilities. Although some tests are applied beforehand, thoroughly prepared tests should be performed. Recently, many free software projects such as language systems and operating systems have utilized test cases, and we too may implement them.

A framework is required to implement these tests, which may cover different types of tests as follows.

The simplest test is the unit test that confirms the elementary features individually in sequence. EPICS software now has the feature "make runtests" to collect the existing test cases. Users can provide tests in the Perl/Test framework.

In order to find combined anomalies that cannot be detected using unit tests, the regression test should be performed. While it is difficult to collect efficient test cases, some of the real running applications with fake data may serve as regression tests.

It may also be necessary to try stress tests because some components such as networks sometimes exhibit unusual behavior under extreme conditions. We may have to install failure-recovery features or failover facilities.

We often find failures just after the shutdown period; further, it is difficult to develop the tests since the hardware and software components are enabled individually in sequence. During shutdown, new software or hardware could be installed, the restoration of hardware or software could fail, or power-cut may create further problems.

Thus, we may require an intelligent procedure to test every feature of hardware and software after shutdown. At KEKB and Linac this is implemented as written procedures developed with observation by human operators, which can be partly automated.

Redundancy

Redundancy may be the last measure to improve reliability, since it is expensive. Nevertheless it is useful not only for failure recovery but also for uninterrupted maintenance.

In any case we may have to prepare backup installations; then, automatic failover is close. There are several possible cases where redundancy is useful.

Currently, RAID and mirror disks are commonly used, and redundant file servers can be employed for network-wide file services. At KEKB and Linac we have employed several different solutions for this purpose since more than ten years ago. We occasionally had unpleasant experiences, but it often helped us during the operation. Furthermore, it simplified the scheduling and maintenance to a large extent.

The redundancy of networks is also an established technology. Before the KEKB project, we employed redundant Ethernet transceivers for more than 40 optical Ethernet links, and obtained favorable results. Nowadays, the combination of the rapid spanning tree algorithm and HSRP or VRRP protocols for higher availability has enabled reliable switches and routers.

The use of redundancy in PLCs is gaining momentum. There are several possibilities regarding the devices that should be redundant, such as power supplies, CPUs, network interfaces, backplanes, and even I/O.

The EPICS redundant IOC is the on-going project at DESY for XFEL. It comprises an RMT (redundancy monitor task), which monitors the robustness of the controllers and manages PRRs (primary redundancy resource), and a CCE (continuous control executive), which synchronizes the internal states between two IOCs. It is necessary to modify the PRRs, which includes scan tasks, channel access server tasks, sequencers and drivers. Users may modify user-supplied drivers and user tasks. KEK recently joined in the project for wider applications with OSI support. ATCA implementation may be possible in the future.

Most of the upper layer servers at Linac are redundant, and this is often beneficial. There should be more areas where software redundancy and replication are useful.

SUMMARY

EPICS and SAD/SADscript have made KEKB a great success, although other accelerators have different criteria. Accelerator control design requires a balance between many aspects, some of which are investigated. There are many effective technologies that have not yet been utilized. Some of them have been examined. Under practical situations, reliability features are required and there exist many possible solutions. Since the control system interfaces with all other facilities in the accelerator, it is the most important. If we have some "phronesis" or practical wisdom, we can solve our problems.

REFERENCES

[1] N. Akasaka *et al.*, "KEKB Accelerator Control System", Nucl. Instrum. Meth. **A 499**, 2003, p. 138.

[2] I. Abe *et al.*, "The KEKB Injector Linac", Nucl. Instrum. Meth. **A 499**, 2003, p. 167.

[3] K. Furukawa *et al.*, "Accelerator Controls in KEKB Linac Commissioning", *Proc. of ICALEPCS99*, Trieste, Italy, 1999, p. 98.

[4] K. Akai *et al.*, "Commissioning of KEKB", Nucl. Instrum. Meth. **A 499**, 2003, p. 191.

[5] `<http://acc-physics.kek.jp/SAD/sad.html>`.

[6] `<http://www.icalepcs.org>`.

[7] K. Furukawa *et al.*, "Timing System Upgrade for Top-up Injection at KEK Linac", *Proc of EPAC2006*, Edinburgh, UK, 2006, p. 3071.