WHAT'S BEHIND AN ACCELERATOR-CONTROL SYSTEM?

Rüdiger Schmitz, Deutsches Elektronen-Synchrotron DESY, Hamburg, Germany.

Abstract

A control system has a lot of features, some essential: e.g. a set of application programs. The infrastructure they need in order to run so that the operators at least be able to switch the accelerator on and off.

Graphical User Interfaces, intelligent control algorithms or data acquisition methods are obvious, but other features (not as obvious) also require considerable manpower and should not be underestimated. They have a major impact on the availability of the control system. I call these features the ‘meta-control system’.

This paper describes the efforts made by the control systems group at DESY to provide a reliable tool for the operators, minimizing the downtime caused by control system failures. It reviews this aspect of computer based accelerator control dating back to the late 1970s when the accelerator PETRA went into operation, controlled entirely by mini-computers from Norsk Data [1].

Both the computer with the supporting technology and the control system group are essential to an accelerator’s success.

INTRODUCTION

MCS -the machine control group at DESY- has built, maintains and improves the control systems of all current DESY-accelerators: The preaccelerators LINAC II, DESY II and PIA, the light sources DORIS III and PETRA III and the free electron laser light source FLASH II. Since the decision to switch off the proton-lepton collider HERA II in 2007, DESY changed its scientific profile from a predominantly high-energy physics laboratory to a synchrotron light research centre. This had a major impact on the required reliability and availability of the control systems:

- The top-up mode for PETRA III does not tolerate any failure in the accelerator-chain for more than 5 minutes.
- The cramped schedule of the beam line experiments at DORIS III, PETRA III and FLASH II may well leave behind an unhappy user if part or all of the requested beamtime is lost.

OPERATOR VIEW

A control system is most visible at the operator-console. Nowadays this is an assembly of monitors and input devices such as mice, knobs or keyboards connected to computers. The operator console is the place from which all available functions of the accelerator in its different phases of operation can be controlled: user run, maintenance periods and machine studies.

The technical implementation differs a lot from control system to control system, but nevertheless the look and feel is not much different. (FLASH and PETRA have different ‘control systems’ but for some areas like vacuum and sequencer there is hardly any difference.) Differences arise from the different age of the accelerators and also from the skills and preferences of the constructor and operator.

Application programs in operator consoles may be rich clients written in Java and Visual-Basic or they may be operator panels generated for example by jDDD and web-based-applications running in a browser such as Web2C [2].

At the other end, there is the accelerator which will be directly affected by actions initiated at the operator-console or by automatic processes running independently of operator interaction. The diagnostic- and machine-protection systems will necessarily report any malfunction of control system procedures.

In between we have what I call here a ‘communication cloud’, i.e. something allowing communication between the operator console and the accelerator. This leads to the simple operator view of an accelerator control system shown in Fig. 1.

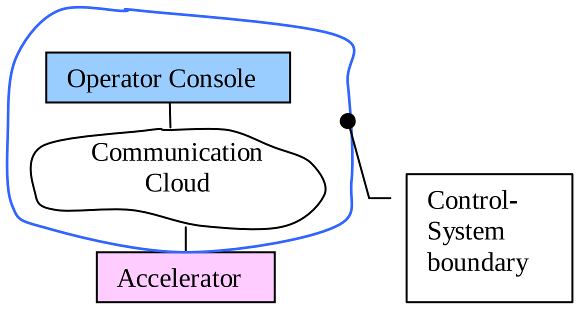

Figure 1: Simple operator view of a control system

CONTROL SYSTEM PEOPLE VIEW

Looking more deeply into a control system one can identify the different hardware building blocks in the ‘communication cloud’: Computers, networks, field buses, diagnostic systems and turnkey systems.

There is no precise definition as to where control systems boundaries are drawn. What belongs and does not belong to the control system is defined in different ways by different people. But at least one needs all major subsystems interfaced to the control system.

An even deeper view will bring us to the software. But at this point the system cannot be understood without yet further information, information which cannot necessarily be found in the control system.

To get information about the underlying principles and concepts of a control system one should ask the control system group. They will use a lot of buzzwords or abbreviations and will refer to documentation web-sites or talks given at conferences. They will use the term ‘control system’ in two different ways:

(1) A system controlling an accelerator, e.g. ‘The control system for PETRA III’
(2) A name for a set of tools providing communication protocols and services which make for efficient client and server applications; e.g. TINE, DOOCS, EPICS, TANGO [3]

I prefer the use of the term ‘control system’ as defined in (1). Running a control system involves more features than are laid down in the documents mentioned above. Fig. 2 illustrates this fact:

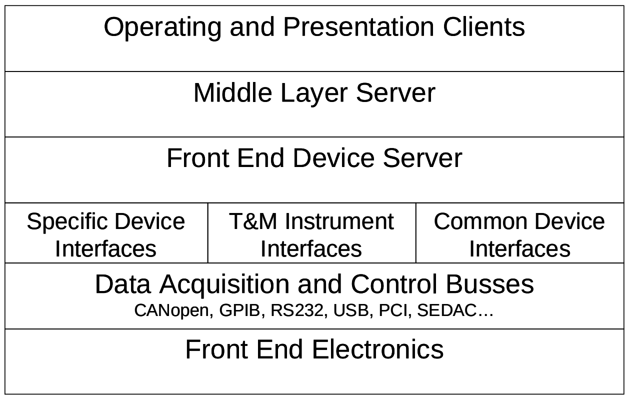
Operating and Presentation Clients
Middle Layer Server
Front End Device Server
Specific Device Interfaces
T&M Instrument Interfaces
Common Device Interfaces
Data Acquisition and Control Busses
CANopen, GPIB, RS232, USB, PCI, SEDAC…
Front End Electronics

Figure 2: Architecture of the control system for PETRA III [4]

Why is Fig. 2 incomplete? A few examples:

- The daily operational needs may require minor or major improvements, perhaps even adding to or redesigning a certain feature. This may lead to pragmatic solutions not foreseen in the description of the tool-sets.
- There are old systems which, due to financial and/or manpower bottlenecks cannot be upgraded.
- The control system has to be able to cope with failures and unforeseen situations.
- The technical and personal environment of the control system changes.
- The diagram shows the system under normal operation conditions, but not the process of achieving these conditions (e.g. when the system is installed) nor the maintenance operations required during the lifetime of the accelerator (e.g. when hard- or software is replaced).
- Some purely pragmatic modifications might lead to a situation in which the diagram no longer represents reality.

Over and above the technical changes just outlined, the control system people are faced with a number of jobs to do and problems to solve. I describe these jobs as the ‘meta-control system’.

META-CONTROL SYSTEM VIEW

Here I describe some of the features that a meta-control system should have. The focus is on the control systems of PETRA III and DORIS III and its preaccelerators, using TINE as an integrating tool-kit.

Fault detection and repair

The following features or improvements have been added or made since 2005 [5]:

- JAVA-Applications write Logging-Information. The control group at DESY has a Log-Viewer-Application to identify faulty applications. Until now, however, there is no automatic notification. (jDDD has recently set a notification to a Java Message Service Server.)
- Remote control of Device-Servers shows status and allows restart.
- What we have named ‘Spider’ shows the status of all links to TINE network devices. A ‘Tarantula’ crawls through those links from one level down to the next, building a tree of dependencies.
- For each type of Windows host used in the control system a spare is kept running and there is an agreed procedure for how to replace a broken computer with the appropriate spare. The time needed for the replacement is about 30 minutes.

Control systems central database

Measures are in place to make our system resilient to disk crashes or computer breakdowns

The configurations and initialisations of Device Server Computers as well as the processes they host are laid down in a Central File Repository. There are semi-automatic procedures using this information to setup a new or replace a broken computer. Work still needs to be done in order to ensure support for different operating systems with the same Central File Repository (e.g. proper choice of file-transfer modes, OS-independent file-formats).

At regular intervals a central upload process copies local files to a network repository. From there they will be downloaded during the setup process mentioned above.

Application deployment

We have implemented a ‘build and deploy’ procedure for JAVA applications [6] which eases the work of the programmers and enforces our guidelines for the creation and storage of JNLP files. (An offline-tool for re-checking the files has yet to be integrated into ‘build and deploy’.)

Application programming policies

A rich client application written for example in JAVA can make use of all features of the language; but we have rules which restrict usage of the features and it is the programmers’ responsibility to obey them. So the control-people may never lose sight of – or allow others to lose sight of – these policies.

An application-framework [7] helps to create applications which are policy conform, and operator panels generated by a panel editor such as jDDD can fully implement the policies.

Maintenance strategies

The control group uses the same operating system updating procedures as the DESY central IT, but provisions have been made for the control group to trigger the process only when it fits into the accelerator schedule.

Proactive maintenance is achieved by looking for critical events in system-log files (DISK-errors, unscheduled reboots). Preventive maintenance involves replacing equipment before end-of-life.

Defence against attacks

DESY's central IT-Group provides and maintains the anti virus software used by the Windows PCs of the control system networks. These networks have no or only limited access to the internet. Access-lists in the control system network routers which could help to protect the control system are under construction.

Access to accelerator equipment is secured by a device-server specific list of ip-addresses and accounts.

Monitoring the system

We recently implemented monitoring tools servers which are fully integrated into the tool-set TINE. e.g.: a **locator service** shows the location of all network-devices connected to the control system-network. This will automatically trace roaming equipment such as vacuum pumping stations. A **network analysis service** gives information on bottlenecks or bad network connections and may generate an alarm.

Integration into Campus IT-Infrastructure

We try to use as many central services as possible. Some services have been introduced by us in close cooperation with the central services people. We have to keep an eye on how the control system is affected when any of the above is out of order.

This strategy has been positive both for us and for the machine-physicists and service-people who easily exchange data between the control system and their workplace.

Observation of hard- and software life cycles in relation to the accelerator's lifetime

A regular review of control systems is advisable, keeping an eye on software- and hardware-lifetimes and on possible improvements or necessary renewals.

The end of support for hard- and software forces us to make decisions. For example Microsoft Windows XP extended support ends in 2014, thus we will not and do not need to upgrade the control system of DORIS III to Windows 7 because the operation ceases by end of 2012.

Choice of adequate hard- and software-solutions

We mainly use standard PCs in the control system, and only occasionally avail ourselves of more expensive solutions. We have generally had success with this policy. It saves money and keeps diversity low.

The chosen hard- and software solution must meet the requirements and the skills and experience of the control system people.

Preserving approved concepts

In 1978 we operated PETRA I with mini-computers. We had implemented many of these meta-control system features. They disappeared along with the computers and, at least some, out of the heads of the colleagues! I think good concepts and their principles should be preserved.

CONCLUSION

An accelerator-control system should support the reliable operation of an accelerator in all its different operational phases with as few interruptions as possible.

The control systems group is responsible for that job, formulating and activating the concepts, policies etc. which hold the control system together and defending it against various quick fix pseudo solutions, which are so often proposed. Indeed, the control system people are the custodians of the meta-control system!

I believe that, in any institution, you will have at least as many control systems as there are control-groups, even if there are no or only slight technical differences. The control systems for DORIS III and PETRA III for instance are technically quite different but are maintained by the same people and the meta-control system is therefore the same.

So what is behind an accelerator-control system? The control system group!

REFERENCES

[1] Norsk Data, Oslo, Norway, ceased to exist 1992
[2] http://jddd.desy.de http://web2ctoolkit.desy.de
[3] http://tine.desy.de http://www.aps.anl.gov/epics
http://doocs.desy.de http://tango-controls.org
[4] From: R.Bacher, "The New Control System for the Future Low-Emittance Light Source Petra 3 at Desy", Proceedings of EPAC 2006, Edinburgh.
[5] P.Duval U.Lauströer R.Schmitz, "Fault Identification in Accelerator Control", Proceedings of PCaPAC 2005, Hayama, Japan.
[6] A.Labudda, "Building and Deploying loosely coupled Console Applications", Proceedings of PCaPAC 2006, Newport News, USA, p 126
[7] K.Hinsch and W.Schütte, "MstApp, a Control Application Framework at DESY", to be published in 2011. See also: J.Wilgen, "First Experiences with a Device Server Generator for Server Applications for Petra III", Proceedings of PCaPAC 2008, Ljubljana.