CONTROL SYSTEMS FOR NEW LARGE EXPERIMENTS

J. Dedic, M. Plesko, R.Sabjan*, I. Verstovsek, K. Zagar, Cosylab, Ljubljana, Slovenia

Abstract

We discuss control systems of accelerators and similar projects that are presently still in design and early construction phases, such as FAIR [1], ESS [2], MedAustron [3], NSLS II [4], ITER [5], etc, and comparing them against the approaches of the last two decades and explain the new trends that are emerging:

- From the organizational perspective, control system architectures are established earlier in the project, allowing them to adapt to the machine physics requirements better as well as allow for modeling and simulations.
- In software, there is much less emphasis on custom codes than there was in the past. Instead, standard and off-the-shelf components and frameworks already used at existing accelerators are becoming the preferred choice, not only reducing risks, but also allowing for reuse and sharing.
- In hardware and networks for real-time control and data acquisition, there is a strong trend from custom electronics development to standard and off-the-shelf solutions. This in particular applies to systems like timing, machine protection, BPMs and LL RF. When custom solutions are needed, flexible hardware technologies (e.g., FPGA) are chosen to allow for future extensibility.

INTRODUCTION

Building a control system for a large experiment has always been a difficult task which required dedicated effort from a big group of people. And we have to thank controls groups in accelerator and the rest of big physics communities for their great achievements.

Control systems evolved in the recent decades, together with information technology, computer science and electrical engineering. In the starting days, little equipment, be it either software or hardware, was available off-the-shelf. A handful of physics labs with difficult requirements, for which solutions have never been implemented, were just not commercially interesting. This led to lots of custom work in the labs. From custom IO board development to advances in computer networking and developing whole software frameworks, nothing was taken for granted. Engineers were also scientists.

During the years, big number of experimental projects and the advance of computing allowed widespread standardization of components. Standard technologies are applied in every aspect of a modern control system, some systems can even be bought completely and some, which are only based on standard technology, but still require a lot of work before installed and commissioned. We shall look at some examples from the current experiments on which we collaborate.

At the end we shall try to summarize and find trends and consequences of progress. The main question is whether the everyday work of controls groups has changed and what does this mean for the main priorities that need to be set at the beginning of every project.

STANDARDIZATION IN LIGHT SOURCES

Plenty of light sources were built in the last decades and they have a lot common with respect to the control system. Control system packages (e.g. EPICS [6] or TANGO [7]) have matured through collaboration and can be easily deployed. They are supported on multiple standard hardware platforms (PC, VME, PXI etc.) and operating systems (Linux, Windows, Unix, Macintosh etc.). They provide solutions for most of your needs. Infrastructure applications like archiving, alarm handling or error logging are provided together with GUI builders and interfaces to many programming languages. Usually, even more than one implementation exists.

Increasing market has attracted industry as well. High performance electronics, made specifically for experiments' requirements is available off-the-shelf. Not only chips, but complete systems like digital BPM electronic [8] or timing systems [9] can be bought. Many equipment or subsystem vendors provide control system drivers with their products and they offer to implement them for the control system package of your choice.

Project leaders and funding agencies know this as well – control system budget has typically fallen from 10% to 5% of the machine's budget (not counting the building and beamlines). The challenge today is to implement a control system with state-of-the-art technology, but with a smaller budget and/or on a shorter time-frame, not sacrificing quality, of course. This prioritizes organizational aspects of the project which will be discussed in later sections.

PUSHING THE LIMITS OF CONTROL SYSTEM COMPONENTS

Other experiments (we have recently worked with ITER, FAIR, ESS and MedAustron) are still hiding more technical challenges and questions. Some examples are explained below.

Machine Protection System

One such example may be a complicated timing system or very flexible, but still safe machine protection system.

---

*rok.sabjan@cosylab.com

Reference (or just similar) implementations do not exist yet and these components are key to success of the whole project.

We have collected requirements from several projects and apart from the traditional role of the machine protection system (MPS) just statically reacting to digital inputs, new features are required. One such is a reconfigurable IO matrix, where responses to interlock inputs would be based on the current mode of the machine. This enables bypassing certain faults or threshold levels. Integration with the timing and control system is highly desired, allowing for quick reconfiguration of the system.

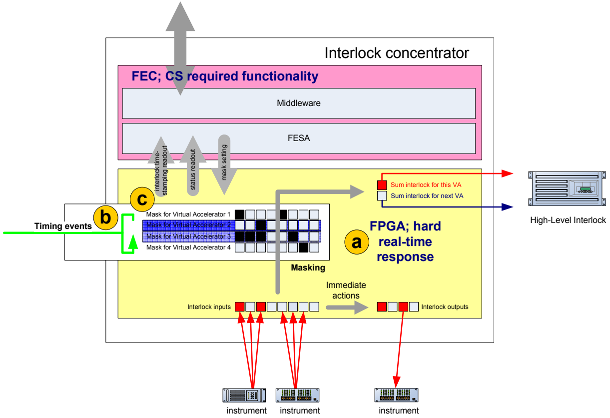


Figure 1: Possible implementation of a fast machine protection system

Integration with the timing system is important for the post-mortem analysis as well. Input signals can be accurately time-stamped and the proper timeline of a problematic event can be reconstructed even if there are a lot of interlocks firing.

Most of today's MPS implementations make use of PLC technology, which has a response rate in the range of several milliseconds. The new design allows response in the range of microseconds even with fibre lengths of over 1 km, making the speed of light the biggest constraint.

Such a standard solution does not yet exist, but the collaboration with a number of labs and their interest makes it worthwhile to start the development. It is our view that the solution to MPS can be a good mix of common general system with specific.

Hard Real-Time Feedback System

Another interesting control system component is a hard real-time feedback system, which brings distributed dimension to the real-time control. Implementations of this already exist and work well (e.g. fast orbit correction for storage rings).

However, current implementations are largely based on proprietary technologies like reflective memory (RM), or are implemented in-house using specialized solutions such as dedicated fiber network and custom hardware. We believe, in order to really standardize on open standards and to lower the cost and the risks for the future, (ten-) Gigabit Ethernet should also be considered.

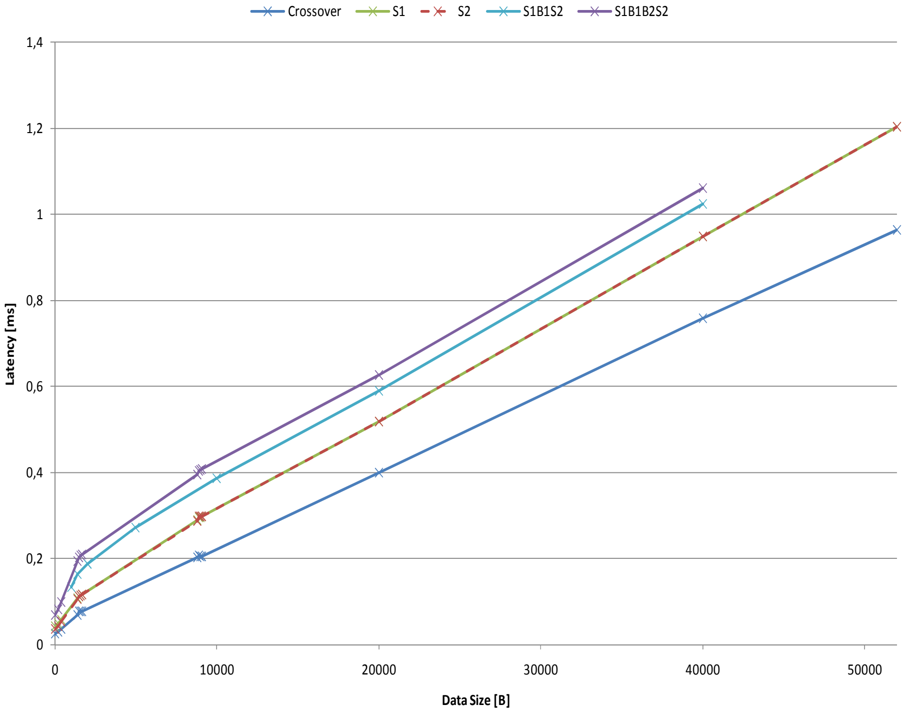


Figure 2: Measurement of latency using RTnet [10] and Gigabit Ethernet as a function of data size. Different lines represent different network topologies, from using a crossover cable only, to complicated topology, where four network switches are used.

We have measured the deterministic performance of Gigabit Ethernet as a task for ITER. We were interested in achieving 1 KHz feedback cycle (2 network hops per cycle) with very low jitter (less than 10 usec) with total traffic of 40kB per cycle. For this, we did not just look at standard UDP packets with multicasting over the network, but we also tested our setup with Xenomai [11] real-time Linux kernel and RT net, a real-time network stack implementation.

Our results showed (Figure 2) [12] that we can already achieve today a very good latency of 0.5ms for data rates that are typical for accelerators. Although we cannot use Gigabit Ethernet technology for ITER requirements today, we are very close. With 10-Gigabit Ethernet just years away, we are confident that Ethernet will be a very good choice for development efforts [13]. Commercially, no other technology can come close – consumer switches, network adapters and cables could be used. It seems very unlikely that this will change in the coming years.

Timing system

New complex machine require a timing system which is more complicated than just a simple event system that is usually used at light sources. New features like virtual accelerators, timing super-cycles (Figure 3) and event acknowledgements are introduced.

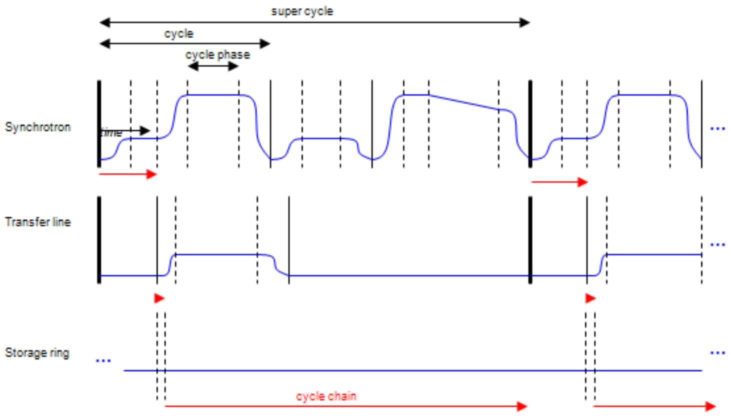


Figure 3: Example timing sequence for FAIR

The existing (off-the-shelf) timing solution like the one produced by Micro Research Finland, which is the most widespread among new machines, cannot provide all the needed functionality, but they can be used as the basic component, the transmission layer.

We see that despite having a commercially available standard solution a lot of customization work is necessary. You can purchase the transport layer, whereas the application layer is machine specific and needs to be implemented for every project individually.

COMPLEX COMPONENTS AND INTEGRATION

We have established that there are definitely trends towards standardization of control system components, which could mean that work is reducing for the controls team. But unfortunately, not everything is that simple.

Components are getting more complex and they require more time and effort to be integrated into your control system. Choices need to be made early in the project which is risky if not all aspects are considered.

Basic control system package

Traditionally the first choice is about the control system package itself (EPICS, TANGO, FESA, TINE, COACK, DOOCS, ACS etc). But this choice is not the most important one. In fact, we believe that people decide for a control system package in a similar way as when they are buying a car: we decide based on emotions and later we rationalize this discussion with architecture description and features. Luckily, most of control system packages are mature and modern technology will enable you to finish your project whatever your choice might be. That is why we recommend choosing the package that you like the most, either due to your personal experience, your people background or because a similar project already used it and those people can help you when you get in trouble.

Integrating other packages

Many facilities use more than just one control system, either they are dealing with a legacy system from a previous experiment, with a component developed by another group or buy machine components with existing commercial control system (e.g. NI LabView [14] or any other SCADA system). Typically, facility control (e.g. air condition) is already automated and needs to be integrated in the main system.

Usage of many different packages is to be avoided, if possible. For the remaining case, I believe that the main control system group must clearly define responsibilities and approve requirements for the interface, especially if other groups are involved. Documentation and maintenance of the systems must be considered. Technical problems come second to interpersonal relationships in this case.

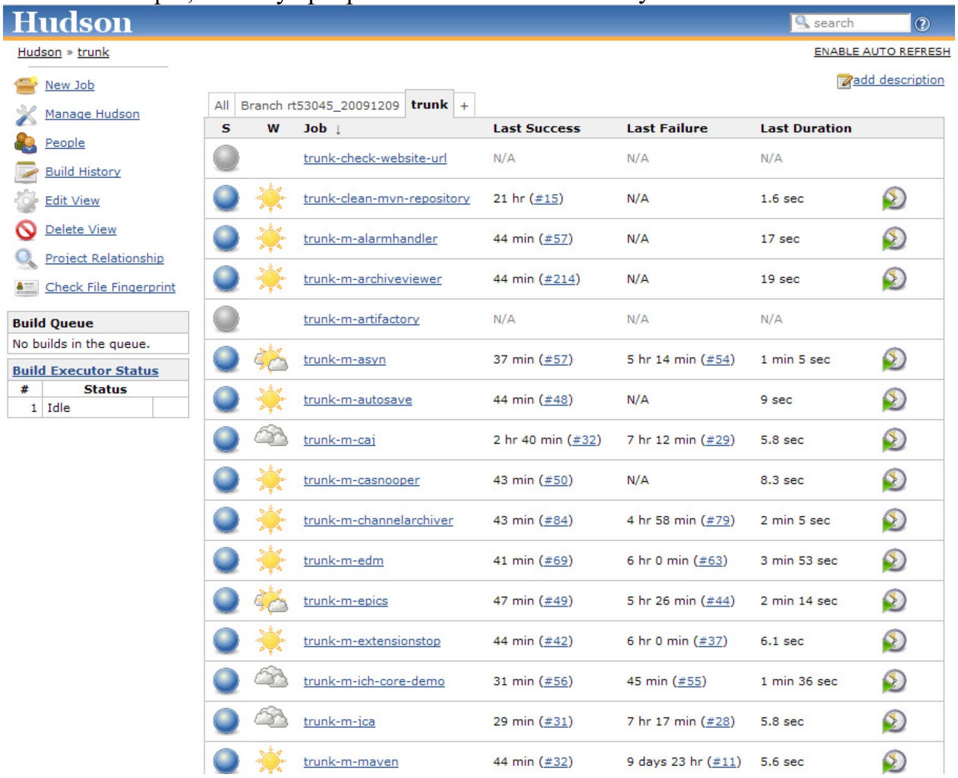
Another set of examples come from the machine physics world. There different packages are used, MatLab [15] and XAL [16] are the most popular recently. Issues here are all the interfaces to other control system components (process variables with all the attributes like alarms, relational database, event handling etc.).

Determining the level of integration is the most important issue to resolve. We need to realize that we are not just "pushing" the data from one system to another, but we must also think about configuration management and maintenance. For example, usually people have different views about which system will check values for alarm levels and how where these thresholds be defined.

In such cases it is usually best to adopt best practices developed and lessons learned by a previous similar experiment.

Distributed development and 'in-kind' projects

New large experiments, such as ITER, FAIR or ESS, are very costly and are often started as international projects with in-kind contributions. The extreme example of this is ITER, where more than 150 plant systems will be provided by the 7 collaborating countries together with the local control system that will be integrated into the main control system.



Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle

S	W	Job ↓	Last Success	Last Failure	Last Duration
○		trunk-check-website-url	N/A	N/A	N/A
●	☀	trunk-clean-mvn-repository	21 hr (#15)	N/A	1.6 sec
●	☀	trunk-m-alarmhandler	44 min (#57)	N/A	17 sec
●	☀	trunk-m-archiveviewer	44 min (#214)	N/A	19 sec
○		trunk-m-artifactory	N/A	N/A	N/A
●	⛅	trunk-m-asyn	37 min (#57)	5 hr 14 min (#54)	1 min 5 sec
●	☀	trunk-m-autosave	44 min (#48)	N/A	9 sec
●	☁	trunk-m-caj	2 hr 40 min (#32)	7 hr 12 min (#29)	5.8 sec
●	☀	trunk-m-casnooper	43 min (#50)	N/A	8.3 sec
●	☀	trunk-m-channelarchiver	43 min (#84)	4 hr 58 min (#79)	2 min 5 sec
●	☀	trunk-m-edm	41 min (#69)	6 hr 0 min (#63)	3 min 53 sec
●	⛅	trunk-m-epics	47 min (#49)	5 hr 26 min (#44)	2 min 14 sec
●	☀	trunk-m-extensionstop	44 min (#42)	6 hr 0 min (#37)	6.1 sec
●	☁	trunk-m-ich-core-demo	31 min (#56)	45 min (#55)	1 min 36 sec
●	☁	trunk-m-jca	29 min (#31)	7 hr 17 min (#28)	5.8 sec
●	☀	trunk-m-maven	44 min (#32)	9 days 23 hr (#11)	5.6 sec

Figure 4: ITER Core System [17] is a software product helping to standardize and ease the development of the control system. To ensure the quality, the software is heavily covered with automatic unit tests which are run at every build. Continuous build system notifies the developers of build problems and test failings within a few minutes.

ITER is tackling this issue with very rigid standardization. Every year, the ITER controls group publishes the Plant Control Design Handbook (PCDH), which describes all the standards, and releases the Core System software (Figure 4), the set of all standard, ITER approved, community tools and software drivers.

The standardization does not stop with the main architecture, hardware platform and IO boards, operating system and software packages. Project life-cycle, naming convention and test plans are also specified.

In addition to this, the Core System software package is prepared. It is the practical aspect of the PCDH and will be used by all ITER collaborators, making it easier to develop the control system properly and easily.

We have recommended this approach to ESS as well and they have adopted the Control Box concept [18]. ESS will also be built by many partners, albeit not as many as ITER.

FOCUS ON DEVELOPMENT PROCESS

Building a complex system from more or less standard components is an engineering task (much more than a scientific experiment) with all the steps that are common to all engineering disciplines. In fact, control system development has an even more complicated cycle:

- Write specifications
- Architecture
- Design

- Prototyping – probably the only fun part
- Define test procedures
- Implementation (coding) – the only software part
- Writing documentation
- Testing (follow ISO procedures)
- Debugging
- Acceptance at customer

Projects are increasingly aware of the development processes. Especially, the international efforts recognize this and focus heavily on the following things. One such is the signal list. It is a golden list that represents the contract between different subsystems and different developers. This is very obvious and should be made in the initial stages, but many projects do not have it until very late in the project.

Signal list also requires a good naming convention, which is unique and still people-friendly. Different people need to access process variables in the control system and naming convention should help not hinder that.

Control groups are putting procedures in place that deal with changing signal list, hardware and software in manner that all interdependencies are taken care of and changes will be applied in all the appropriate places. This avoids project inconsistencies.

There are two more important procedures that are considered: logistics of installation and error handling (i.e. bug fixing). How one handles control system installation and testing needs to be defined well before integration time, even before any outsourcing contracts are written. It should define what are the necessary testing steps before integration starts, who is responsible for what part and what are the interfaces between different groups of people (control system people, device experts, subcontractors, electrical support team).

We all accept that some bugs are inevitable and sufficient time needs to be planned for testing and debugging. The procedure should also define how bugs are reported and how changes (fixes) are introduced and re-tested. Last by not least, good development practices minimize the number of bugs in the first place.

Big projects realize that man-power is a problem and it is difficult to cover the wide range of required competences. That is why they decide for outsourcing for a big part of control system, whereas they retain the overall system responsibility in-house.

CONCLUSIONS

Standardization is the key trend emerging with development of new and complex projects. Labs are not required to develop all parts of a control system themselves, but can rely on re-using development from other people or even buy off-the-shelf components and solutions. Technical risks are reducing.

Today, integration is the biggest aspect of a controls project. How will all the components fall into the main architecture, what will be the interfaces and how any of the requirements will be addressed, are the main questions. Integration starts with day one and is an every-day companion throughout the project.

Organizational risks in big and complex project with many partners are increasing. Focus needs to be shifted to stricter definition and implementation of development processes and rigorous standardization with clearly defined interfaces.

In short, control system development is becoming more and more an engineering discipline and less like a science.

REFERENCES

[1] FAIR; http://www.gsi.de/fair

[2] European Spallation Source; http://www.ess-scandinavia.eu

[3] MedAustron; http://www.ebgmedaustron.at

[4] NSLS-2, National Synchrotron Light Source 2; http://

[5] ITER, ; http://www.iter.org

[6] EPICS collaboration; http://www.aps.anl.gov/epics

[7] TANGO collaboration; http://www.tango-controls.org

[8] Instrumentation Technologies; http://www.i-tech.si.

[9] Micro Research Finland; http://www.mrf.fi

[10] RTnet: Hard Real-Time Networking for Real-Time Linux; http://www.rtnet.org

[11] Xenomai: Realtime Framework for Linux; http://www.xenomai.org

[12] K. Zagar et al, "Evaluation of High-Performance Network Technologies for ITER", 7th Technical Meeting on Control, Data Acquisition and Remote Participation for Fusion Research, Aix-en-Provence, June 2009

[13] K. Zagar, "Ethernet-based Real-time Networks for Distributed Cloosed-loop Control", PhD Thesis, to be published

[14] National Instruments; http://www.ni.com

[15] MatLab CA; http://ics-web.sns.ornl.gov/kasemir/mca

[16] XAL; http://www.ornl.gov/~t6p/Main/XAL.html

[17] K. Zagar et al, "ITER control system development environment", this conference proceedings.

[18] T. Satogata et al, "ESS Controls Strategy and Control Box Concept", this conference proceedings.