THE DO's AND DON'Ts IN PROCESS CONTROLS LESSONS LEARNED OVER 35 YEARS

M. Clausen, T. Boeckmann, J. Hatje, O. Korth, M. Moeller, J. Penning, H. Rickens,
B. Schoeneburg, Deutsches Elektronen Synchrotron DESY, 21220 Hamburg, Germany

Abstract

Designing, implementing and maintaining process control systems for cryogenic plants requires different viewpoints compared with those in machine controls. 24/7 operations for more than a year is a basic requirement. Hardware and software must be designed to fulfil this requirement. Many projects are carried out with industrial partners. Companies specify the process control logic which gets implemented by the local DESY team. Responsibilities, time tables and milestones must be clearly defined in such a case. Several cryogenic installations have been equipped with state of the art process control systems for cryogenic controls. Where the last one being the European XFEL. In the course of time commercial and open source systems were implemented and maintained. Control loops were basically always implemented in front end controllers running the real-time operating system VxWorks and EPICS as the control system toolkit. The approach to use PLCs will be discussed as an alternative approach. Large installations like the European XFEL require good project planning. Our success story will finalize our look back and initiate our look forward.

PROCESS CONTROLS AT DESY

In 1982 process controls for cryogenic systems was implemented in hardware PID controllers. Only a few engineers had the knowledge how to operate such a system. A failure over night was a night mare because no diagnostics were installed.

Over the years archive systems were installed. Even alarm systems found their way into cryogenic control systems because 24/7 operations required immediate action if some conditions were suspicious.

Over the years all of the cryogenic processes (cryogenic plants and cryogenic distribution systems) are controlled by process controllers. Some of them went through two basic refurbishments. In the end all cryogenic process controls at DESY and the XFEL are implemented in EPICS front end controllers – so called Input Output Controller (IOC).

This paper will describe a subset of the experiences we gained of the years. Namely: PLC integration; testing of new equipment and project management issues.

TO USE OR NOT TO USE PLCS IN PROCESS CONTROLS

PLCs can be really useful – there is no doubt. The question is: 'Where should PLCs be used?'

PLCs for Machine Interlocks

A very prominent usage for PLCs is the area of hardware interlocks. In former times this kind of hardware protection was implemented in hard wired logic. This logic slowly moved into intelligent controllers and finally into PLCs. These interlocks are well defined. They are thoroughly tested and should not be altered. Implementing this logic in a PLC is a no-brainer and used basically for every system in place.

PLCs as Data Concentrator

The next occasion where PLCs find their way into control systems is the usage as a data concentrator. There are a lot of different I/O signal types which can be connected via specific signal-conditioning modules to PLC type of communication controllers. Depending on the vendor theses are PLCs with I/O modules or intelligent communication controllers with I/O modules which can be programmed like a PLC.

In both cases the controller or the PLC will function as a data concentrator and will not be used for control functions in the field. Typically these controllers will be connected to a field bus like Profibus or CAN or (real-time) Ethernet.

Communication with the Process Controller

The communication between the data concentrator and the process controller will run through one of the field busses mentioned above. Therefore the process controller has to implement the necessary driver for that type of connection.

The communication must be configured for each individual I/O signal as if the signal would be connected to the process controller directly. This kind of configuration will be typically limited to one channel for each I/O signal. Basically this is a one to one representation.

Reliability

Reliability is a strong argument in favor of PLCs. For sure PLCs are known for their reliable runtime behavior. They run in thousands of instances and the code is (should be) thoroughly tested. All of the hardware components are nearly mill proof and reliable as well. Crashes of PLCs are not known to the author. If the same kind of application is run on an EPICS IOC one would have to put the same constraints on software and hardware on the EPICS implementation. A Windows or Linux operating system will hardly reach the level of reliability which a PLC based OS will provide. A 'real' real-time operating system like VxWorks is tuned for reliability and will work differently from OS's which require for instance hard

drives. The structural blocks of a PLC controls logic are not custom made. These are designed for reliability. The same implementation on an IOC would run as reliable as long as basic control blocks are used. Custom made EPICS 'records' or subroutines can easily cause distortions in the runtime behavior. Last not least the hardware requirements on an IOC must be comparable to those of a PLC. PLCs do not use the most powerful CPUs which need active cooling. They also come without hard drive. 'No moving parts on a reliable IOC'. This is a string requirement on reliable IOC.

Taking the above requirements into account there's basically no good reason why an EPICS based IOC should not run as reliable as a standard PLC. Examples show EPICS IOCs which have run for several years without interruption.

Redundancy

If the requirements on reliability go even further it is possible to run PLCs as well as IOC in a redundant setup. Both implementations are available.

Flexibility

Before discussing flexibility one has to agree on the setup being used. In this discussion the setup will consist of a PLC, an EPICS IOC and a console application on a PC. The next condition is the location of the process controls software. Where does the logic reside? On the PLC side or the IOC side?

Control Software on the PLC

A very obvious location could be the PLC. The logic will be 'close' to the I/O and take advantage of the reliable implementation of the PLC. The implementation of the controls logic will be straight forward because many companies specify their requirements in the controls logic in 'sort of' PLC language. Thus the implementation in the PLC is nearly 1:1. Once the logic is implemented it is a strong requirement that _all_ of the parameters in the controls logic shall be made available up to the operator console. This implies that the data exchange between PLC and IOC must be configured in a way to make that happen. Even basic control block in a PLC can be pretty complex. Between 30 and 60 different properties are possible. In addition precautions must be taken to allow not only for read but also for write operations. This way one control block on the PLC side may result in about one hundred records on the IOC side.

Besides the obvious overhead in the sheer number of records to be defined the configuration- and change management adds another level of complexity.

Last not least runtime diagnostic and runtime access to the controls implementation must be taken into account when control software shall run on the PLC. A simple example might explain the difference: If the controlled value of a control loop shall be changed from one sensor to another it would be a no-brainer on an EPICS IOC and is possible from the console level. Even if the sensor resides on another IOC it would still be possible to change the record name online. There's no way to do that on the fly on a PLC based implementation.

Control Software on the IOC

The arguments on the previous topic already describe the disadvantages of PLC based implementation in which we ran at DESY. The flexibility of running process control software on IOC or IOC type of controllers is the basis for running cryogenic controls at DESY and now for the European XFEL for the last 25 years. 24/7 operations require the maximum flexibility wih process controllers permanently running. Loading new software during operation is not an option. The existing implementation must provide all the flexibility the operators need to survive unforeseen situation. Any process value with all it's properties must be available on the console level without prior configuration. Operator or at least process engineers must be able to manually change (nearly) any property in the control system to continue operation by any means.

Besides this flexibility requirement one has to discuss the effort to configure both systems. In the latter case the I/O must be configured in the data concentrator or communication processor or PLC. This is the same in both approaches. The difference is the communication between communication processor and the IOC. In this case only the basic data including status and error conditions must be exchanged between both sides. This can be implemented in a well-structured way. The implementation of the controls logic would use record structures which are similar to the PLC block structures. Any property of these so called EPICS records will be available by default. No configuration will be necessary. In addition it is possible to add so called sequence- or state notation programs in a high level language. Implementing process control logic on the IOC side has clearly many advantages.

Process Engineer and Controls Engineer

Having mentioned the advantage of process control software on the IOC side it is still not always obvious that this is the only 'correct' approach. In the end it comes to people: Their experience and practical and theoretical background. A controls engineer with a strong PLC background who gets involved with process engineering will have the tendency to implement the controls software on the PLC side. A process engineer with no experience on the implementation of controls software might have the tendency to choose the more flexible approach.

The DESY/ XFEL Approach

DESY has a long history in implementing process control software on IOC type front end controllers. Initially a commercial control system was in place. Later that was replaced by EPICS IOCs. Reliable IOCs are running on Compact PCI CPUs which are powered by redundant power supplies and diskless without active cooling. Many of the IOCs are running redundantly. On the other hand we run so called 'soft IOCs' on virtual Linux machines implemented in a Hyper-V cluster on a set of 4 powerful computers. – The best of both world one may say. Of

course DESY also runs PLCs. But mostly for hardware interlocks. Some PLCs though run 'black box' software from the vendor of the controlled hardware. This is not always the best approach.

TESTING CONTROLS APPLICATIONS

A thorough test of controls applications requires a very elaborate implementation of test code. This can easily reach a level of complexity which is higher than the software which shall be tested. Especially testing the dynamic behavior of the controlled process can be very time consuming and will consume resources which are typically not available in this phase of a project.

Before thinking of the best of all worlds – namely dynamic testing – the test environment should cover the very basic requirements.

Test Environment: Simulation of Input Channels

A very basic requirement is to be able to simulate the input channels of the system. This can be implemented by setting the input channels into 'simulation mode'. Another approach is to let the input channels read from dedicated simulation channels instead of the connection to the 'real' I/O hardware (which will not be available for software commissioning). In both cases two tools should be configured:

Save/ restore for the Simulation Channels

Since we are living in a real world changes to he software will be necessary. These changes will probably require restarting the software. Procedures have to run through again (and again). To ease this process it will be useful to store the intermediate values in a save/ restore tool. This way an intermediate state can be recovered easily and the throughput of the test procedure will be more productive.

Graphical User Interface for all Channels to be Tested

Synoptic displays should be available for each channel from the very beginning. This is important to display not only the value but to get access also to all of the other properties of the channel. The preparation of groups of 16 channels using so called 'faceplates' is a well-established procedure at DESY. Faceplates for each type of I/O signal and different types of control loops have been created. They are configured by scripts in groups of 16.

Full Integration Test End to End

A final test from the sensor to the operator panel – or synoptic display should always be carried out. This final step will ensure that the data path from the sensor to (PLC to) IOC with all the necessary configurations the control system channel names and the graphical representation of the values – like STRING or DOUBLE values – will be tested and verified.

Archiving for Later Evaluation of the Test Results

Setting up the archive system from the very beginning is a goody but not absolutely necessary. Is is useful for bookkeeping and creating archive plots for the final acceptance protocols.

Setting up Alarms

Some signals will generate alarms which shall be used for soft-interlocks. It is useful to have the alarms configured upfront for checking the results in the alarm retrieval tools.

Logging in State Notation Programs

State notation – or sequencing programs do not per se offer logging of the status and states. It should be implemented from the very beginning to get messages especially on state changes. These should be grouped into messages just for commissioning and other messages which will be used for the operators during normal operations. Commissioning messages should be disabled or deleted after commissioning to avoid message floods.

PROJECT ORGANIZATION

Working in a scientific organization implies that the final design of equipment is shifted to the last minute. This opens the chance to provide the best technical design to the users. Besides the positive aspect of providing the latest/ greatest technical design it surely also has its drawbacks.

What kind of lessons could be learned from such an approach? The experience shows that working on the final design of the control system – and finishing it - before the final design of the controlled equipment is defined will cause a lot of frustration. Changes are necessary in the control system on the hardware side as well as the configuration and even the synoptic displays. Two approaches are possible to overcome frustrating experiences.

Just In Time Development

It is possible to wait with the final design of the control system until the final design of the controllable components has been approved. This will make sure that you implement what is really necessary. All the necessary control channels will be available but just in time does not necessarily mean – just in time –ready-! Waiting until the last minute could cause problems implementing the control logic and synoptic displays in time. You'll run into a phase of the project where everybody is extremely busy and necessary experts to specify the software might not be available.

You might end up with a control system which is barely working but does not perform the way expected. You might think you optimized the workload because no work had to be done twice but working in the last minute causes a lot of overhead in terms of personal stress and coordination effort between all parties involved.

In the end a system will be in operation which performs 'just OK' but not good. It will cause a lot of frustration and might even cause problems during commissioning and the first phase of operating the new equipment. There is an alternative approach:

One Step Ahead Development

For the last big project – the European XFEL - we have chosen a different approach. We tried to calculate our workload precise enough from the very beginning. This implies that the requirements and specifications especially on the hardware-side must be 'frozen' way before controls equipment gets installed in racks. This even differs more when the installation takes place by external companies. A well-defined milestone for the final electrical layout of the control system must be specified. This is the milestone called: 'If you meet that milestone we'll meet the commissioning milestone'.

Commissioning Controls Hardware

Regardless where the controls racks are built they will need up to three checks:

1. A factory acceptance test will make sure that the completion of the controls hardware will meet about 95% correctness in terms of mechanical and electrical installation.
2. A pre-installation check. This will be a full integration check with a fully configured control system. The controls logic will be in place and the so called database (in EPICS speak) will be configured. So called Faceplates will be available for each I/O channel. Basic graphic panels will allow checking the full functionality of each channel. Ideally also the synoptic display will be in place in order to check the signals from the hardware signal up to the operator console.
   At this level also the archive system should be configured to provide the tools for cross checking the test results.
3. Last not least all channels will be tested when the controls racks are installed in the final destination. Once this test is passed the controls are ready for machine operations.
   For a cryogenic system it means that the cryogenic operations can be started. Valves can be operated and all readout signals are available.
4. Special cases in cryogenic systems are level measurements and heater controls. The final test of these systems can be performed as soon as the first liquid is 'dropping into' the cryogenic system. Special precautions should be taken to start reading level signals and running heaters not before liquid temperatures have been reached.

The biggest advantage of this procedure is that step one and two can be carried out way before the real installation in the field takes place. It is essential to know that the controls equipment has run through a 100% signal test. As soon as the 'OK' for installation is available you know that the expected error rate on the I/O system will be less than one per mill.

Quality of Cable Works

To reach the one per mill error rate from the sensor to the operator panel you'll have to make sure that the cabling will be as reliable as the electronic racks themselves. In our case we were lucky that just one team worked on the cabling and carried out this task throughout the whole installation procedure. It should not be under estimated that getting one or more teams familiar with the cable plans for your specific layout might require working time from the controls team and might be error prone if teams are changing regularly. The one per mill error rate can only be achieved if the cabling team does _not_ change on a regular basis. Otherwise higher error rates and time consuming corrections will be necessary.

Sliding Project Plans

No project ever will stick to the initial time table and project plan. Of course the project management will always argue that milestone are carved into stone and might not slide but we are living in a real world…

How can milestones and sliding project plans be implemented side by side? This is much easier than it might look like. Implement your project plan like an 'If – Then – Else' statement in any programming language. It could look like this: If the milestone 'Installation of control racks' is kept we will need x-weeks until we are ready for machine commissioning – else – our project plan will 'slide' by (the same number of) x weeks.

This way you always keep the necessary time for the controls commissioning independent of the starting date. You might have to add constraints like 'during summer time from A to B it will take C-weeks longer due to vacation of part of the team etc.

As a result one should avoid defining the end of controls commissioning as a milestone. Since controls are always the last ones to finish – by definition – the time for preparation will get squeezed into the remaining time available and – in the end - it will not be enough to finish the task. Project management might ask why controls are late and all those things which should be avoided.

If there's a message from the experience of getting several projects implemented it is this one:
Implement a project with a minimum of hard coded mile stones and get especially controls' project plan implemented as a sliding project plan – based on previous mile stones.

Working with Industry

Working with industries can be fun – if well-defined rules have been established. It is not a secret that many companies (hopefully not in general) are happy about any change in design and delay caused by the customer especially if the customer has to provide deliverables like controls software. To avoid unnecessary discussions it is advisable that project plans are clearly defined and do not open points for interpretation or discussion.

It could easily happen that deliverables by the customer are delayed because the boundary conditions and specifications are missing. Such a conflict can be avoided if the conditions are clearly described. Like: The controls software will be ready for commissioning x-weeks after the requirements and specifications have been delivered to the customer. After commissioning y-weeks will be necessary to incorporate the changes into the software. After this the controls software will be ready to commission the equipment and to run later acceptance tests.

This is a variant of the sliding project plan. The controls software will not be ready at a certain date (milestone) but after well specified conditions have been reached. In many cases these conditions are under responsibility of the industrial partner. As soon as this is understood you will reach a well-defined partnership.

CONCLUSION

Working for 36 years at DESY was interesting from the first day and except a few occasions I enjoyed working here every day. Being part of a team which built HERA in the 80^s and later the XFEL in the 2010s was a great experience which I never want to miss.

Results count: The quality of the XFEL cryogenic controls implementation is speaking for its own.

ACKNOWLEDGEMENTS

I want to thank the colleagues in the cryogenic controls group for the fruitful collaboration and joint work over 33 years at DESY. My thanks also go to the engineers and operators of the cryogenic machine group at DESY. Also the good relationship to the DESY machine controls group should be mentioned.

I want to thank everybody in the EPICS community for their excellent developments. Without the highly reliable software from the EPICS community it would not have been possible to finish the tasks at DESY and the XFEL.

Last not least I want to thank my wife Marlies for her permanent support.