CERN PROTON SYNCHROTRON COMPLEX HIGH-LEVEL CONTROLS RENOVATION

S. Deghaye, M. Arruat, D. Garcia Quintas, M. Gourber-Pace, G. Kruk, O. Kulikova, V. Lezhebokov, S. Pasinelli, M. Peryt, C. Roderick, E. Roux, M. Sobczak, R. R. Steerenberg, J. P. Wozniak, Z. Zaharieva, CERN, Geneva, Switzerland.

Abstract

After a detailed study of the Proton Synchrotron (PS) complex requirements by experts of CERN controls & operation groups, a proposal to develop a new system, called Injector Controls Architecture (InCA), was presented to and accepted by the management late 2007. Aiming at the homogenisation of the control systems across CERN accelerators, InCA is based on components developed for the Large Hadron Collider (LHC) but also new components required to fulfil operation needs. In 2008, the project was in its elaboration phase and we successfully validated its architecture and critical use-cases during several machine development sessions. After description of the architecture put in place and the components used, this paper describes the planning approach taken combining iterative development phases with deployment in operation for validation sessions.

INJECTOR CONTROLS ARCHITECTURE

With most resources focused on LHC systems, the implementation of the Java applications in the PS complex was done as a short-term solution with minimal impact on existing infrastructure. No re-engineering of the architecture or the infrastructure was undertaken and the integration with new controls components was kept to the minimum. As the maintenance cost is increasingly higher and the integration of new features difficult, it was decided to setup a team of controls and operation experts to study the PS complex requirements in terms of high-level controls [1]. The result of the working group was the proposal of a new system called Injector Controls Architecture (InCA) [2]. Aiming at the homogenisation of the control systems across CERN accelerators, InCA is based on components developed for the LHC but also new components required to fulfil the specific operation needs of the PS complex. The InCA proposal was presented to and accepted by the management late 2007 and the system is now under development since the beginning of 2008. The first operational target is set on the PS machine for the first part of 2010. However, an early deployment of a simplified system was already successfully conducted for the 2009 run of the Low Energy Ion Ring (LEIR).

Overall Architecture

InCA is based on the now classic 3-tier architecture that offers several advantages compared to the 2-tier solution in production today. Amongst them, one can cite better performance thanks to centralised computing, better scalability and something that becomes more and more important, better security.

Figure 1 gives an overview of the different components involved in the project and their corresponding place in the control system.

In the lower tier, one finds the front-end computers dedicated to the real-time control of the accelerator hardware. Software-wise, this part is covered by the Front-End Software Architecture (FESA) [3]. The InCA renovation project is only concerned by the public interface of FESA towards the high-level control system.

In the middle tier, one has the components implementing the main accelerator high-level services (1) the Control core based on the LHC Software Architecture (LSA) [4] (2) the Acquisition core based on the newly developed acquisition component, called AcqCore, for the scalar parameters and the Open Analogue Signal Information System (OASIS) [5] for waveform acquisitions and finally (3) a configuration service for the configuration information retrieval.

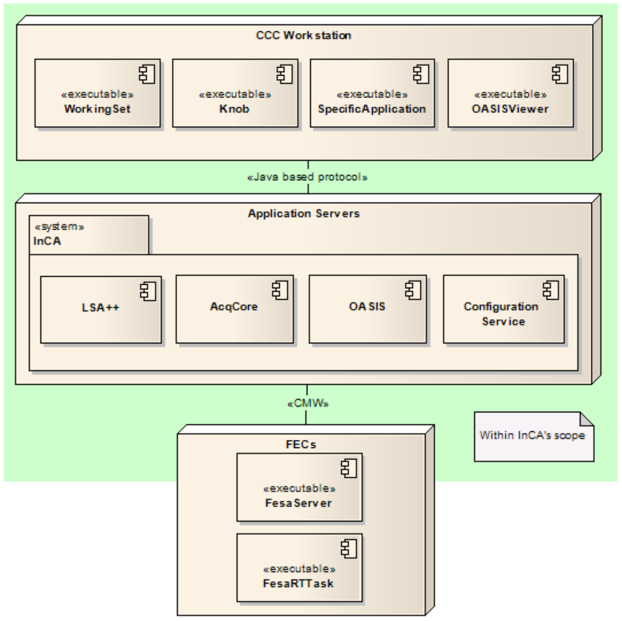

Figure 1: The main components of the Injector Controls Architecture (InCA).

In the top tier, running in the CERN Control Centre (CCC) workstations, one has the Graphical User Interfaces (GUI) made of generic applications to observe and control the accelerator devices (e.g. knob, function editor...) and, whenever the processing to be done at application level requires it, the specific applications such as the steering application YASP (Yet Another Steering Program) [6].

CONTROL CORE

The Control Core component deals with all the services that involve accelerator settings. The main services are the parameter trimming, the setting archiving and the parameter reference storage. After a thorough study of the LSA system put in place for the LHC, it was decided to base the InCA Control Core on an evolved version of LSA supporting features required by the PS complex.

LSA has, amongst other things, a powerful settings management system. It is driven by a database where a description of the machine elements, along with optic values, and parameters are stored [7]. LSA supports the declaration of high-level parameters (e.g. the angular deviation produced by a dipole corrector) thanks to parameter hierarchies that relate the different control parameters between each other. Several modifications have been made to the system in order to fit the requirements of the PS complex. The noteworthy extensions are the introduction of bi-directional parameter hierarchies where the propagation can be done from the low level to the high level (from the hardware to the model) and not only top-down. This allows the users to choose whether they work with a fixed model (top-down) or a fixed machine (bottom-up). We have also added or improved support for types such as the function lists and enumerations. In addition, a new settings archiving system is under development. Technical details on LSA can be found in several articles such as [4].

ACQUISITION CORE

The acquisition part of InCA is split into two components - the AcqCore and OASIS. OASIS is purely dedicated to signal observation, either analogue or digital, in the time domain. More details on OASIS can be found in [5]. The AcqCore implements the parameter acquisition and parameter status services.

The parameter acquisition service focuses on providing the operation crew with the latest acquired values of the hardware devices in a timely manner. The parameter status service computes and publishes information indicating the status of the accelerator. It is based on a comparison algorithm, specified by the operation group, using the control and acquisition parameters' values. The rationale behind the creation of the Acquisition Core is twofold. First, we wanted to reduce and stabilise the traffic between the control room applications and the real-time front-end computers (FEC). Just for the PS machine, there are up to 12'000 parameters to be processed every 1.2 second and, in 2007, we suffered from performance issues that obliged us to quickly put short term solutions in place. Even though the FECs will get more powerful in the future, they are still embedded computers with limited resources that must be spared as much as possible. Besides, the concentration of the acquired values in a central node gives us the possibility to compute high-level acquisition parameters' values that could not be done in the FECs because they do not have access to the information (e.g. the machine optic). This feature will help to reduce the number, or at least the complexity, of the specific applications.

Technical Description

Figure 2 gives an overview of the architecture put in place for the AcqCore. All data coming from control devices is temporarily stored in the bounded values buffer. This data is processed by a number of tasks running concurrently and driven by events. The set of rules (Life Cycle Policy) defining how a particular task should react on event is assigned for each task configured in the system. For example, for the data processing task, the policy declares that the task should be started at the beginning and finished at the end of each accelerator cycle performing several attempts to process the data.

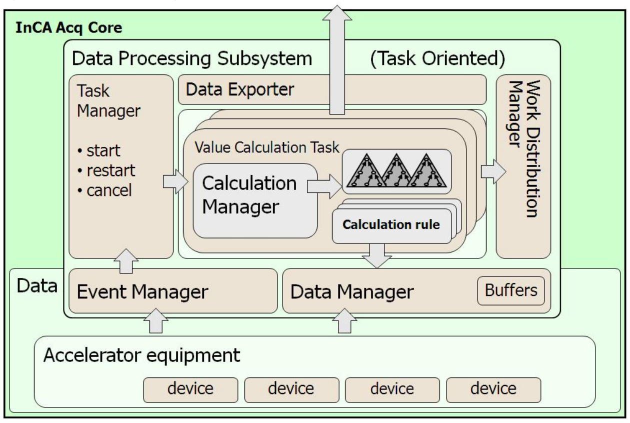

Figure 2: Acquisition Core overview.

Data processing implies calculating values of high-level parameters based on a number of parameter values received from devices (low-level parameters). This process depends on the parameter hierarchies which define the relations between parameters and the rules to be used to calculate parameter values (see Control Core). Once the values are calculated, they are sent to the clients (typically GUIs) using a grouping mechanism via Java Messaging Service (JMS).

CONFIGURATION SERVICE

The configuration service is about providing the other components with information on the control system (device/parameter description, application layout...). Due to the large amount of information required, the configuration service must be implemented with high performance in mind.

In InCA, the service is located in the middle tier to ensure scalability and performance. It is planned to use the requests made to the configuration service to predict and anticipate the tasks to be performed by other components (e.g. subscription preparation), therefore speeding up the processing further.

GENERIC APPLICATIONS

In order to support basic operations with the control system, we have developed a handful of generic applications that allow routine operations to be easily performed on any device. For device control, we have the Knobs and the FunctionEditor acting respectively on scalar or function parameters. For system overview and problem detection, there is the WorkingSet application that presents, in tabular form, sets of devices along with their key values and statuses. The generic applications rely completely on the middle tier services: configuration service for the layouts, acquisition core for the acquisitions and control core for the trims.

SPECIFIC APPLICATIONS

Even though most of the basic operations can be done with the generic applications described in the previous section, there are cases where a dedicated application is required. We have currently a few tens of such applications and we will not detail them here. The application that received most of our renovation attention last year is the Automatic Beam Steering application (ABS) that will be replaced by the steering application already in use on several machines (SPS, LEIR & LHC).

Steering Application

YASP, which stands for Yet Another Steering Program [6], was originally developed for the automatic steering of LHC beams. This application relies heavily on LSA services such as element optics and high-level parameter control. As LSA is also used as the control core of InCA, the decision to generalise YASP to all CERN accelerators was obvious. After population of the database with machine layout, optics and calibration curves, several test sessions were performed to ensure the new corrections are as good as the ones provided by the old system. Some corrections, such as the low energy orbit correction, are already at an operational stage but for other corrections (e.g. transfer line steering) there are still some specifics to be addressed before YASP can replace the old X/Motif ABS.

ITERATIVE DEVELOPMENTS WITH REGULAR MACHINE DEVELOPMENT

The developments are organised using a light version of SCRUM [8]. The development cycle is represented in figure 3. Every milestone is a machine development (MD) session where the PS machine is reserved for our tests for 6 to 8 hours. During the MD, InCA is deployed and tested full scale on the operational accelerator. We perform mainly scalability and performance tests.

The milestones are separated by a period of 3 or 4 months during which we develop a coherent set of new features to be validated during the next MD. Within the milestone iteration, we perform 3 to 4 week long development iterations, known as sprints in SCRUM. As we focus on working software, all development iterations are ended with a demo of the new working features in front of the operation crew representatives. Before the MDs, we foresee at least 3 weeks for the stabilisation phase where all the components are integrated and tested altogether. This cool down period is of paramount importance not to waste any precious MD time. Our experience shows that, even though we try to keep the system continuously integrated, a stabilisation phase is required in order to fix potential integration and deployment issues.

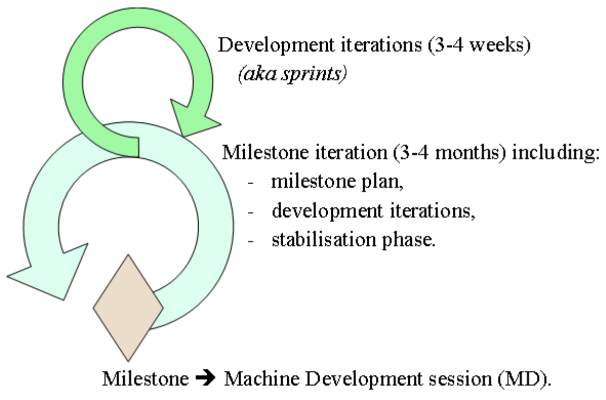
Development iterations (3-4 weeks)
(aka sprints)
Milestone iteration (3-4 months) including:
- milestone plan,
- development iterations,
- stabilisation phase.
Milestone ➔ Machine Development session (MD).

Figure 3: InCA development cycle.

CURRENT STATUS & FUTURE PLANS

Early 2009 it was decided to deploy a simplified version of InCA on the Low Energy Ion Ring (LEIR). This simplified version had all the InCA components previously described except the AcqCore that was replaced by direct FEC subscriptions. We successfully validated the generic applications in operation. The next goal is set for mid 2010, just after the non-LHC physics resume in the PS complex. The plans are to deploy the first version of InCA on the PS machine. The state of the developments indicates that the set of critical features required by the operation group to deploy version 1.0 of InCA is well on track.

REFERENCES

[1] M. Benedikt *et al.*, "Strategy for PS complex control software renovation: Working Group Mandate", https://edms.cern.ch/file/849420/1/WorkingGroupMandate.doc (internal publication).

[2] S. Deghaye *et al.*, “Injector Controls Architecture – Vision”, https://edms.cern.ch/file/863516/5/Vision.pdf (internal publication).

[3] M. Arruat *et al.*, “Front-End Software Architecture”, ICALEPCS’07, Knoxville, Tennessee, U.S.A.

[4] G. Kruk *et al.*, “LHC Software Architecture (LSA) – Evolution toward LHC beam commissioning”, ICALEPCS’07, Knoxville, Tennessee, U.S.A.

[5] S. Deghaye *et al.*, “OASIS Evolution”, ICALEPCS’07, Knoxville, Tennessee, U.S.A.

[6] J. Wenninger, ‘‘YASP: Yet Another Steering Program.’’

[7] C. Roderick and R. Billen, “The LSA Database to Drive the Accelerator Settings”, ICALEPCS’09, Kobe, Japan.

[8] H. Kniberg, “SCRUM and XP from the trenches”, http://www.infoq.com.