ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

UPDATE ON THE STATUS OF THE FLUTE CONTROL SYSTEM

W. Mexner*, E. Blomley, E. Bründermann, C. Fehlinger, A.-S. Müller, R. Ruprecht,
T. Schmelzer, M. Schuh, N.-J. Smale, Karlsruhe Institute of Technology, Karlsruhe, Germany
S. Marsching, aquenos GmbH, Baden-Baden, Germany
I. Kriznar, Cosylab, Ljubljana, Slovenia

Abstract

The first phase of FLUTE, a new linac based test facility and THz source, is currently being commissioned at the Karlsruhe Institute of Technology (KIT). It consist of an RF photo gun and a traveling wave linac accelerating electrons to beam energies of 40 to 50 MeV. The control system is based on a virtualized infrastructure running Ubuntu Linux and Linux KVM. As base for the SCADA system we use EPICS 3.15 with Control System Studio (CSS) for the GUI. The long term data storage is provided by a Cassandra NoSQL database. This contribution will present the architecture and the current status of the FLUTE control system.

INTRODUCTION

FLUTE [1] is a new R&D linac accelerator (see Fig. 1) offering beam energies of 7 to 41 MeV for the development of accelerator technology, new diagnostics and instrumentation for fs bunches. It will be used as a test facility for the study of bunch compression with all related effects and instabilities like space charge, coherent synchrotron radiation (CSR) as well as the different generation mechanisms for coherent THz radiation. Furthermore it will serve as a broad band accelerator-based source for ultra-short and intensive THz pulses, e.g. for time- & frequency-domain spectroscopy of kinetic processes.

CONTROL SYSTEM OVERVIEW

The design of the FLUTE control system [2] is based on EPICS 3.15 with Control System Studio (CSS) as the main operators interface to the control system.

The DAQ has been driven by the demand to make systematic studies of the beam compression and the generation of THz radiation. The pulse synchronous (10 Hz) data-acquisition based on a Cassandra NoSQL database has to record for each pulse diagnostic information about RF pulses, laser pulses, pulse charge and of course the overall accelerator settings.

Operator GUI Concept

The top level panels of the operator GUI have an synaptic approach. A 3D model of the Accelerator is used to for easy orientation (see Fig. 2). An error is visually connected to the device and to the position along the machine. The corresponding control panel for the device can be opened by a direct link on the panel. In addition all device panels are included in a list, which can be used to navigate to the devices, grouped by their task. As the accelerator is in its first phase the starting overview panel shows the injector section. For each of the next sections there are also synaptic overview panels planned as well as for the complete machine.

Stepper Motor Control

The FLUTE accelerator and the KARA Beamlines are using the same inhouse developed stepper motor control and driver concept. For motion control, we use the OMS (Pro-Dex) 5 axis Ethernet based MAXNET controllers and as 2 phase motor drivers we use the BCD130.x family of MIDDEX electronic allowing up to 12800 micro steps per revolution. EPICS hardware integration was done with the Pro-dex OMS asyn motor base axis support. For axes using an encoder, we experienced the problem that the motors would sometimes move randomly. We could solve these problems by applying a patch that synchronizes the motor with the encoder position before every movement.

Timing System

The timing system for FLUTE is based on the hardware components from Micro-Research Finland [3]. We use the VME form-factor for both the event generator (EVG) and the majority of event receivers (EVRs). However, we do not use the VMEbus for controlling these boards. Instead, we use their Ethernet interface to control them using a UDP/IP-based protocol.

* wolfgang.mexner@kit.edu

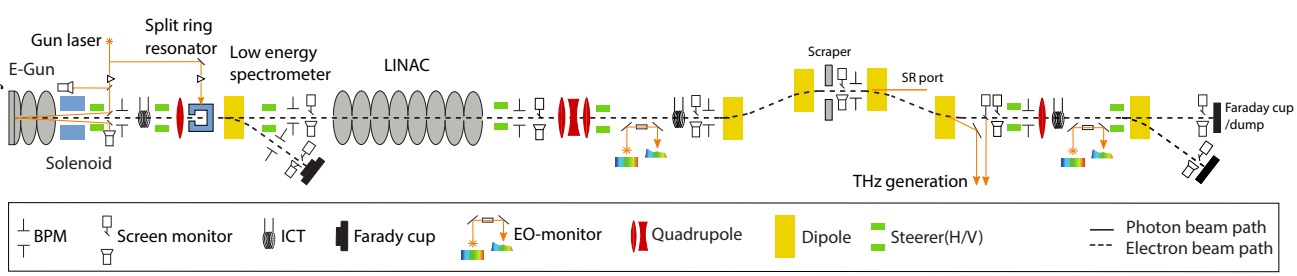


Figure 1: Scheme of the FLUTE accelerator with all installed and planned components [1].

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

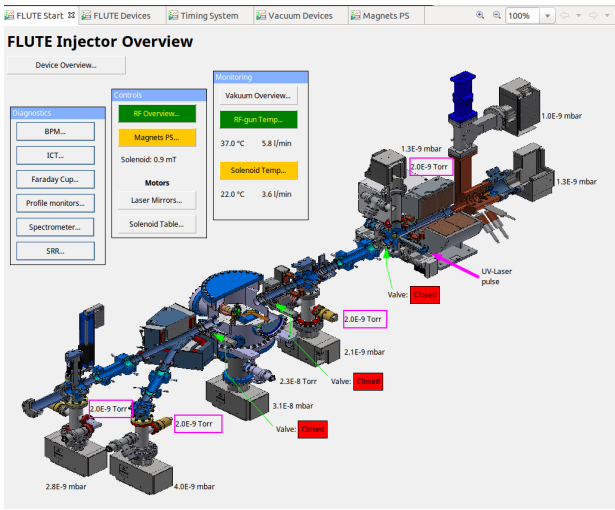


Figure 2: CSS Synaptic Operator view.

As the traditional EPICS device support for MRF devices [4] did not support this way of controlling the devices and its architecture did not include a layer for implementing different ways of transport, we developed our own EPICS device support [5] which includes such an abstraction layer.

Thanks to the flexibility of this new device support, we can essentially use the same code for controlling MRF boards over the network and controlling them locally over PCIe, which we need for some µTCA EVRs. The same device support is also used for the timing system of KIT's KARA facility, where it has proven to work reliably as well.

RF System

The RF uses a pulse type Klystron to produce a 4 µs long 3 GHz burst of 45 MW of power. This power is split between the E-Gun and the LINAC. The power supplies, for charging the Pulse Forming Network (PFN) and controlling the klystron coils are all slow control using PLCs, which are fully integrated into EPICS and CSS. However, MTCA based controllers such as the laser synchronization feedback loop, Low Level Radio Frequency feedback loop (LLRF) , and BPM electronics (readout via MTCA) which were developed for the FEL at DESY are not yet fully integrated to EPICS. As a stepping stone to full integration, these devices are presently used with DOOCS and JDDD installed on the devices. For standard configuration settings a remote desktop is used to access the JDDD panels. For the 'daily operator use', the more used configuration parameters have been integrated to EPICS and CSS using either gateways or Chimera TK adapter (the latter developed in cooperation by aquenos, DESY, HZDR and TUD). These methods are also used to transfer fast data capture to the operator display and database for storage. This is also true for the timing distribution system which, because of historical reasons, have a mix of the MRF VME and the X2 MTCA timing systems.

The software of the LLRF systems is based on the ChimeraTK framework [6]. Thanks to the use of this framework, essentially the same software can be used in control-system environments based on DOOCS, EPICS, or OPC-UA, making it possible to use the same hardware and software in a number of facilities like ELBE, the European XFEL, FLASH, FLUTE, and TARLA with only minor tweaks. This has helped to reduce the man-power needed to commission the system at FLUTE significantly.

SERVER INFRASTRUCTURE

Most services needed for FLUTE run within virtual machines on a virtualized infrastructure. We use the Linux Kernel-based Virtual Machine (KVM) [7] for hosting both Linux and Windows virtual machines (VMs). We use Ubuntu 14.04 LTS as the operating system for the VM hosts and manage the VMs using libvirt [8]. To ensure that basic services like network gateway (firewall) and DNS are always available, these essential services are provided by multiple VMs, hosted on different host systems. These host systems use a dedicated direct link for heartbeats, so that high-availability management is not affected by disruptions of the networking service.

This combination has proved very reliable: This virtualized infrastructure (based on two host machines) has been running for four years while needing virtually zero maintenance. In the near future, we plan to migrate the system to Ubuntu 18.04 LTS so that we can continue to apply critical security updates past April 2019.

We assessed that virtualization is not the best option for services with significant demands on the CPU or memory. For such services, sharing hardware resources does not make sense because it would only increase the demands on the virtualization hosts' hardware, while not profiting from the sharing of resources (which is the primary objective of virtualization in the first place).

For this reason, we also use three Supermicro Microcloud chassis [9], each containing 12 nodes. Each node is equipped with a Xeon E3 processor, 32 GBs of memory and two hard-disk drives. We use these nodes to run services like the Cassandra PV Archiver [10] that provides long-term storage of process data and the service for processing camera images.

NETWORK INFRASTRUCTURE

In order to improve the reliability and security of the network services, we use a dedicated network for all measurement and control equipment of the accelerator. The network is completely separated from the institute's office network. The VM hosts are based in both the dedicated accelerator network and the office network, so that selected VMs can provide services for transferring data between the two networks in a secure, well-controlled fashion. One example of such a service is the EPICS PV Gateway [11] that enables users to access EPICS process variables from their office computers, but limits all access to be read-only, thus ensuring no one can accidentally interfere with the operation of the accelerator.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP10

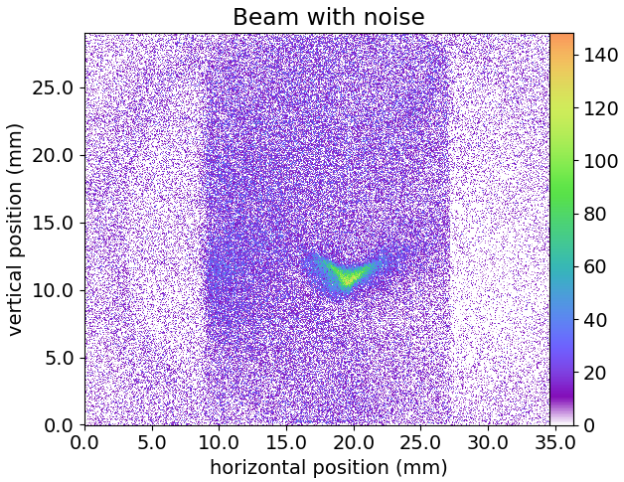


Figure 3: First FLUTE Beam.

CSS CLIENT DEPLOYMENT

The deployment of CSS consists of two separate, but closely related tasks: First, deploying CSS itself (the actual software along with its specific configuration for the environment), and second deploying the panels used inside CSS. The first task could be accomplished through means of the operating system (e.g. packaging and deploying CSS as a Debian package), but the panels change too frequently to make this a feasible approach for them. In addition to that, due to its heritage from Eclipse, CSS works around the concept of a "workspace" and no two instances using the same workspace can run at the same time.

These issues caused us to develop a couple of Python scripts that take care of installing or updating CSS, preparing a fresh workspace on each start, and cleaning up this workspace after CSS is closed. These scripts were originally developed for use at KIT's KARA facility, but they were quickly and easily adapted for use at FLUTE. These scripts are distributed alongside the panels as part of a Subversion repository. When the local working copy of the repository is updated, both the panels and the scripts are updated and thus the version of the panels and the version of CSS that is automatically installed by the scripts always match. By creating a fresh workspace on every start, the operators always start with the same, familiar view (Fig. 2), regardless of which panels have been opened or closed previously. This was a crucial factor to improve the acceptance of CSS as the user interface for operators.

SUMMARY AND OUTLOOK

In May 2018 first electron beam was produced and observed by different diagnostic systems such as the screen monitors (see Fig. 3), beam position monitors, integrated current transformer, and Faraday cup. Based on the recorded data the different systems can be calibrated which is needed for the full integration into the control system. At FLUTE the first user's experiment is foreseen in 2019 within the ARIES Transnational Access programme funded from the European Union's Horizon 2020 R&I programme under GA No 73 08 71.

ACKNOWLEDGEMENT

The authors would like to thank DESY's MSK group and KIT's Institute for Data Processing and Electronics for their support regarding the RF system. They would also like to thank PSI's diagnostics group for their help regarding beam diagnostics.

REFERENCES

[1] A. Malygin *et al.*, IPAC'18, Vancouver, 2018, TPHMF068. A. Malygin, A. Bernhard, E. Bründermann, A. Böhm, S. Funkner, I. Križnar, *et al.*, A. Malygin *et al.*, "Commissioning Status of FLUTE", in *Proc. IPAC'18*, Vancouver, BC, Canada, Apr/May 2018, pp. 4229–4231. doi:10.18429/JACoW-IPAC2018-THPMF068

[2] S. Marsching *et al.*, "Status of the FLUTE Control System", in *Proc. PCaPAC'14*, Karlsruhe, Germany, 2014, paper WPO013.

[3] Micro-Research Finland, http://www.mrf.fi/

[4] EPICS mrfioc2 device support, http://epics.sourceforge.net/mrfioc2/

[5] EPICS MRF device support provided by KIT, https://github.com/kit-ibpt/epics-mrf

[6] M. Killenberg *et al.*, "Abstracted Hardware and Middleware Access in Control Applications", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017. doi:10.18429/JACoW-ICALEPCS2017-TUPHA178

[7] Linux Kernel-based Virtual Machine, https://www.linux-kvm.org/

[8] libvirt Virtualization API, https://libvirt.org/

[9] Supermicro MicroCloud, https://www.supermicro.com/products/nfo/MicroCloud.cfm

[10] Cassandra PV Archiver, https://oss.aquenos.com/cassandra-pv-archiver/

[11] EPICS PV Gateway, https://epics.anl.gov/extensions/gateway/index.php

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP11

ETHERCAT SOLUTION FOR THE TPS CONTROL SYSTEM

C. Y. Liao, C. Y. Wu, Y. S. Cheng, K. H. Hu, K. T. Hsu
NSRRC, Hsinchu 30076, Taiwan

Abstract

The EtherCAT real-time Ethernet technology is now widely used in the field of industrial automation. This paper is to evaluate and establish the EtherCAT (Ethernet for Control Automation Technology) solution for digital and analogue I/O and motion control in accelerator applications. Thanks to developments at the Diamond Light Source, EtherCAT is integrated into EPICS with support for most devices of a general data type. Preliminary tests and plans are summarized in this report.

INTRODUCTION

The Taiwan Photon Source (TPS) is a low emittance, high brightness synchrotron light source, located in Hsinchu, Taiwan. New control technologies have begun to emerge in the past few years and we pay specific attention to the EtherCAT technology [1], which is a fieldbus system based on Ethernet and was developed by Beckhoff Automation [2] in 2003. It has become one of the mainstream communications interfaces for connecting to PLCs, sensors, servo motors, I/O and other automation equipment. The goal of EtherCAT development is to enable Ethernet applications to reduce data update time, lower communications jitter and reduce hardware costs for application to automation. The EtherCAT protocol is standardized in IEC 61158. The EtherCAT master sends a telegram through to each slaves and each EtherCAT device reads the data and inserts its data into the frame as it moves downstream. EtherCAT supports almost any topology, such as line, tree or star.

Several support programs were developed for communication between EPICS IOC and the EtherCAT master. At the Diamond Light Source (DLS) support is being developed based on a scanner process through the IgH EtherCAT which can transfer data between EPICS IOC and EtherCAT master [3] through the UNIX socket. Based on IgH EtherCAT, the PSI [4] also develops an EtherCAT driver, ecat2, for the same purposes. The HE Youngcheng [5] uses the OPC (Object Linking and Embedding for Process Control) gateway driver, which can operate in a Windows OS system. In this study, the DLS-EtherCAT driver is used. It can support most EtherCAT devices of a general data type. A system architecture, plans and implementation will be presented in this report.

ETHERCAT EPICS ARCHITECTURE

Platform

The platform for EtherCAT application is based on the Linux OS, the real-time patch is optional. Two Ethernet ports are basic configuration, one for control system internet and one to connect to EtherCAT devices. In our applications, CompactPCI (Advantech) [6], MXC industrial PC (Adlinktech) [7], UP Squared (Up-board) [8], Raspberry pi [9], all are acceptable. Some existing control systems for ID control are based on the cPCI platform. As a new platform, we select the MXC platform. UP2 and Raspberry are used for special standalone applications. The x86 or x64 works as the EtherCAT master platform.

IgH EtherCAT Master

The IgH EtherCAT Master is an open-source EtherCAT communication software tool [10] for the Linux platform. The devices in the EtherCAT network can be divided into master and slaves. The master acts as the controller of the entire network and is responsible for sending instant packets to each slave to read data or control its behaviour. The Linux OS kernel version must be 2.6 or 3.x. Currently the version of IgH EtherCAT modules is 1.5.2 which was released on 2013-02-12. The performance was studied by Soyeon Kim [11], which shows that the real-time performance is similar to or better than Beckhoff's TwinCAT. The Ethernet driver is divided into the native EtherCAT-Capable Ethernet drivers and the Linux standard Ethernet driver. Users must use the first driver to support real-time communication. After installation of the IgH EtherCAT master, execution of the command "ethercat version" can check if the machine has the EtherCAT driver module loaded, the command "/etc/init.d/ethercat status" checks the status of the IgH EtherCAT master, and the command "/etc/init.d/ethercat stop(start or restart)" controls the behaviour of the IgH EtherCAT master.

DLS EtherCAT EPICS Support

The Diamond Light Source (DLS) EtherCAT is an EPICS support module to interface an EtherCAT bus to EPICS. It uses a scanner process that serves as a server to communicate with EPICS IOC. The PDO (Process Data Objects) and SDO (Service Data Objects) access are supported. The module along with Diamond's Remote I/O was presented in [3]. An EtherCAT device with "REAL" data type of PDO is not supported. We hope to be able to support it in the future. The DLS EtherCAT support consists of four components:

1. Real-time Linux kernel (Optional)
2. IgH EtherCAT device driver
3. Scanner
4. IOCs.

The EPICS environment with asynDriver (asynchronous driver support) and Sequencer (snc/seq) modules are needed for running the DLS EPICS IOC.

Start-up

After the slaves (EtherCAT devices) are connected to the setup, the EtherCAT command line tool can help to visualize it. First, the platform must find the EtherCAT devices

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP11

via the command "ethercat slaves". For example, the TPS IU22 insertion device has EtherCAT devices and this command will list all its devices as shown in Fig. 1. Then, using this information, the DLS-EtherCAT chain.xml file is created to configure the sequence and device names. Compiling (Make) the chain.xml file generates the scanner.xml file and operation of the scanner process with the scanner.xml. The scanner process will open up a UNIX socket waiting for communication with the EPICS IOC, thus starting the IOC process to create the PVs.

```
172.20.3.17:~ # ethercat slaves
 0  0:0   OP  +  EK1100 EtherCAT Coupler (2A E-Bus)
 1  0:1   OP  +  EL1819 16K. Dig. Eingang 24V, 10?s
 2  0:2   OP  +  EL2819 16K. Dig. Ausgang 24V, 0,5A, Diagnose
 3  0:3   OP  +  EL3104 4K. Ana. Eingang +/-10V Diff.
 4  0:4   OP  +  EK1100 EtherCAT Coupler (2A E-Bus)
 5  0:5   OP  +  EL3104 4K. Ana. Eingang +/-10V Diff.
 6  0:6   OP  +  EL3104 4K. Ana. Eingang +/-10V Diff.
 7  0:7   OP  +  EL4134 4K. Ana. Ausgang -10/+10V. 16bit
 8  0:8   OP  +  EL9505 Netzteilklemme 5V
 9  0:9   OP  +  EL1124 4K. Dig. Eingang 5V, 10?s, Sensorversorgung
10  0:10  OP  +  EL2124 4K. Dig. Ausgang 5V, 20mA
11  0:11  OP  +  EK1100 EtherCAT Coupler (2A E-Bus)
12  0:12  OP  +  EL3318 8K. Ana. Eingang Thermoelement (TC)
13  0:13  OP  +  EL3318 8K. Ana. Eingang Thermoelement (TC)
14  0:14  OP  +  EL3318 8K. Ana. Eingang Thermoelement (TC)
172.20.3.17:~ #
```

Figure 1: List of EtherCAT devices installed for the IU22 insertion device.

APPLICATIONS

The DLS-EtherCAT EPICS supports several EtherCAT devices with the general data type. Some of the devices are testing, such as Beckhoff, ADLINK, MOONS, and Taiwan Pulse Motion EtherCAT devices. A wide range of applications can be considered for the accelerator, such as insertion device control, power supply, interlock system, feedback control et al and many devices, including DIO and AIO control, temperature monitor, SSI encoder monitor, and et al are employed.

Insertion Device Control System

For insertion devices, EtherCAT modules are used such as digital and analogue I/O for correction power supply control, gauge pressure reading and temperature monitoring. Figure 2 shows the IU22 insertion device control system layout. EtherCAT devices used are shown in Table 1. The MXC-6300 is used as a control platform with Linux kernel 3.10.0 (rt56) CentOS 7, EPICS base 3.14.12.7, and IgH EtherCAT master 1.5.2.

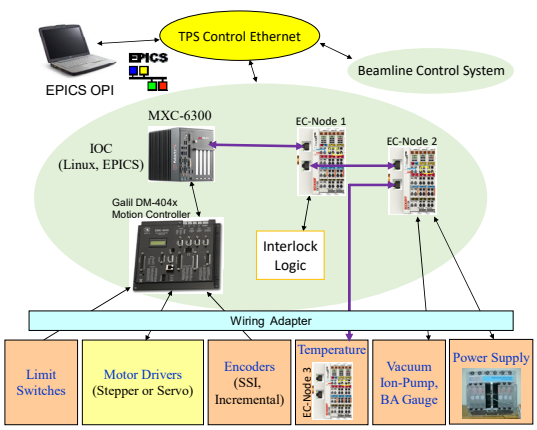


Figure 2: IU22 insertion device control system layout.

Table 1: List of EtherCAT Devices used for the IU22 Insertion Device Control System

	Device	Function
EC-Node 1	EK1100	EtherCAT Coupler
	EL1819	DI for limit switch and subsystem status input
	EL2819	DO for alarm reset and brake open
	EL3104	AI for driver signals monitor
EC-Node 2	EK1100	EtherCAT Coupler
	EL3104	AI for coil power supply monitor
	EL3104	AI for gauge pressure monitor
	EL4134	AO for coil power supply control
	EL9505	Power Supply terminal 5V
	EL1124	DI 5V for coil power supply fault
	EL2124	DO 5V for coil power supply reset
EC-Node 3	EK1100	EtherCAT Coupler
	EL3318	Thermocouple, Ch01-08
	EL3318	Thermocouple, Ch09-16
	EL3318	Thermocouple, Ch17-24

ID Encoder Malfunction Protection

Insertion devices (ID) are essential components in third-generation synchrotron light sources. Reliable operation of insertion devices is important to users of the beamlines. The most unpredictable fault is due to a soft error of absolute optical encoders due to irradiation. There are several solutions to avoid this kind of fault, one is to increase the distance of the encoder from the beam level, the other is a lead cover shielding and the final method is to adopt auxiliary position sensing devices to help recovery from a fault.

The Beckhoff EL3255 (potentiometer reader) module and potentiometer (GEFRAN, PZ-12-S-200 for Gap, PZ-12-S-075 for Phase) are installed in the ID as shown in Fig. 3. After calibration of the potentiometer reading with SSI encoders, the auxiliary absolute position information can be used to cross check the absolute SSI encoder health for malfunction protection purpose. A protection process is developed to compare the position from two sensors, if the difference is too large, the abort motion command will be sent to the controller to stop the motor driver.

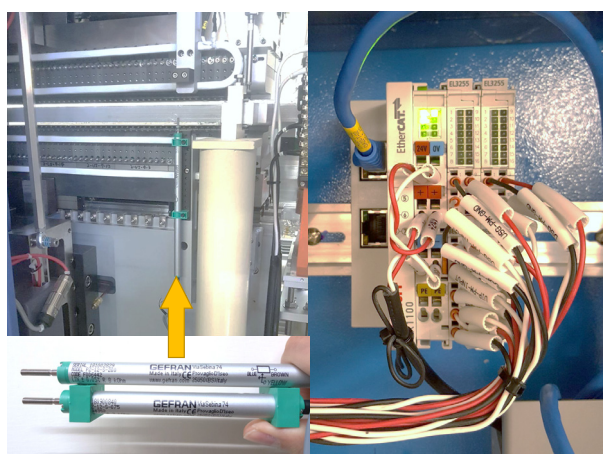


Figure 3: Potentiometer and EtherCAT devices for ID.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP11

SUMMARY

The EtherCAT real-time Ethernet technology is now widely used in the field of industrial automation control. The EtherCAT hardware is introduced into the TPS acceleration control system due to its simple configuration and easy extension. The Diamond Light Source EtherCAT EPICS support is selected for communication with the EPICS environment. It's easy to use and works well with most EtherCAT devices for digital and analog I/O, temperature monitoring, encoder monitoring, and et al. Currently, the new TPS insertion device control system uses EtherCAT devices. Another application in the TPS is to develop a protection process for the insertion device when the optical encoder fails. Thanks to the short cycle time of the EtherCAT protocol, the protection cycle can reach 1 kHz. The features of the EtherCAT in the TPS use the EtherCAT motion controller for motor control applications and adoption of the EtherCAT based data acquisition (DAQ) modules for signal acquisitions are in planning.

ACKNOWLEDGMENTS

The authors are greatly thankful to those who have developed the EtherCAT EPICS support for reading EtherCAT based hardware. And thanks to Ronaldo Mercado (Diamond Light Source) for assistance in resolving the issue of reading/writing the SDO address for this study.

REFERENCES

[1] EtherCAT Technical Introduction and Overview, www.controldesign.com/assets/Media/MediaManager/EtherCAT_Introduction.pdf

[2] Beckhoff Automation, www.beckhoff.com

[3] R. Mercado et al., "Integration EtherCAT Based IO into EPICS at Diamond", in Proc. *ICALEPCS'11*, Grenoble, France, Jul. 2011, paper WEMAU004.

[4] Dragutin Maier-Manojlovic, "Real-time EtherCAT Driver for EPICS and Embedded Linux at Paul Scherrer Institute (PSI)", in Proc. *ICALEPCS'15*, Melbourne, Australia, Jul. 2015. paper MOPGF027.

[5] HE Yongcheng et al., "A method of communication between EPICS IOC and EtherCAT devices", Nuclear Techniques, Vol. 37, No. 11, November 2014.

[6] Advanteck cPCI platform, www.advantech.tw

[7] Adlinktech, MXC platform, www.adlinktech.com

[8] Up-board, Up Squared platform, www.up-board.org/up-squared

[9] Raspberry pi platform, www.raspberrypi.org

[10] Etherlab IgH EtherCAT, www.etherlab.org/en/ethercat

[11] Kim, Soyeon and Euncheol Shin, "A Performance Evaluation of Open Source-based EtherCAT Master Systems", Proceedings of the 4th International Conference of Control, Dynamic Systems, and Robotics (CDSR'17), page 128, Toronto, Canada.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP13

CONTROL SYSTEM UPGRADE FOR THE FFAG ACCELERATOR COMPLEX AT KURNS

Y. Kuriyama*, Y. Fuwa, Y. Ishi, Y. Mori, H. Okita, T. Uesugi
Institute for Integrated Radiation and Nuclear Science, Kyoto University, Osaka, Japan

Abstract

Fixed field alternating gradient (FFAG) accelerator complex has been operated as a proton driver for the experiment of accelerator driven system (ADS) at Institute for Integrated Radiation and Nuclear Science, Kyoto University (KURNS). PLC based control system has been developed and the operator interface has been connected to PLC via network. Originally, a LabVIEW based operational interface was chosen to construct the system because of its easiness. However we met an upgrade problem, and a new control system based on EPICS instead of LabVIEW was introduced in 2010. In the spring of 2018, the replacement from LabVIEW to EPICS has been almost completed except for the beam interlock system and the LINAC control system provided by LINAC production company (AccSys). Also, the EPICS archiving tool (Archiver Appliance) has been invoked and operated at the end of 2017. This presentation reports the details of the current control system and also the upgraded GPIB control and storage system.

INTRODUCTION

FFAG accelerator complex has been operated as a proton driver for the ADS experiment [1]. The construction of this complex had been started since the fiscal year of 2002 and with this complex, the world first ADS experiment was carried out in March, 2009 [2].

The KUNRS complex consists with two ion sources, H^- linac and four FFAG rings. Figure 1 shows the schematic view of KURNS complex.

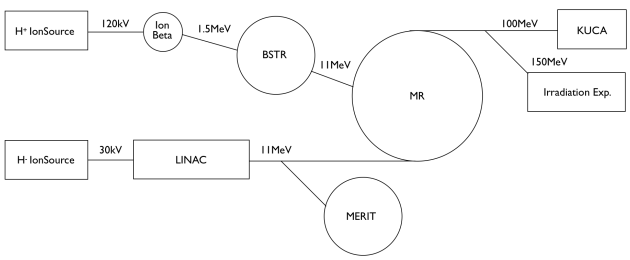


Figure 1: Schematic View of the KURNS Complex.

To control this complex, PLC based control system had been developed. Originally, operator interface had been developed based on a LabVIEW [3]. Since 2010, the aim for the sophistication and stability, we introduced EPICS control system [4]. The immigration from LabVIEW to EPICS had been almost completed in the spring of 2018 except for the beam interlock system and Linac control system.

* kuriyama@rri.kyoto-u.ac.jp

OVERVIEW OF CONTROL SYSTEM

In the KURNS complex, PLC-based systems have been adopted since the beginning of construction. The operator communicates with the PLC via the network, and the control target device such as the electromagnet power supply is operated by receiving a command from the PLC.

About 20 PLC-CPU modules and 20 Linux PCs, 5 Windows PCs and 2 board computers and 2 servers are used. Also, 20~35 network cameras are used for an equipment monitoring and beam profile observation.

These control devices are connected by an independent network for the accelerator control. Figure 2 shows a diagram of control system.

Since all the devices are connected to the same network, high network load is becoming a problem.

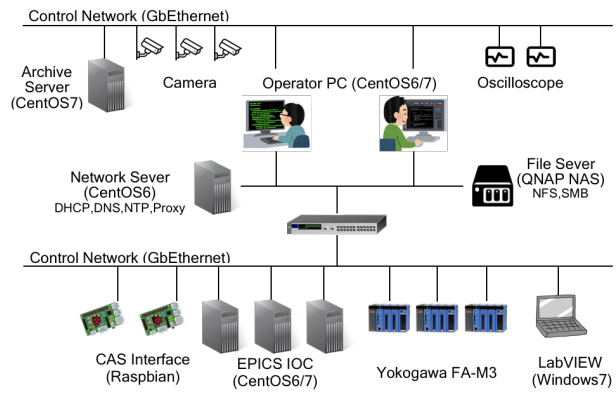


Figure 2: Schematic View of the Control System.

For the device control system, EPICS-based ones and GPIB controlled ones are mixed. EPICS-IOC consists of Yokogawa's PLC FA-M3 and CentOS6 or CentOS7 PC. The number of EPICS-IOC is about 15.

From the point of necessary processing capacity, it is possible to aggregate them into several units. Since accelerator components are often replaced at the KUNRS complex, EPICS IOCs are prepared for each section such as ring and beam line to cope with this. It realizes the flexibility of control system.

RENEWAL OF STORAGE SYSTEM

During the past 10 years, KURNS control system had changed file server three times. One is Linux-based file server and two are mac os based file servers.

Since the period of use was exceeded five years in December 2017, we began to consider updating the file server. Requirements for the new file server are

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP13

- Supported Protocol : NFS v3, SMB 2.0,
- Redundancy : RAID 6,
- Backup : External Storage, Snapshot,
- Ease of replacement : Hot-swappable, General Drive Bay (SATA).

In order to satisfy the above requirements, we decided to shift to a NAS dedicated machine instead of a x86 based PC.

The adopted hardwear is TS-431P2-4G manufactured by QNAP [5]. It has Arm-based CPU, 4 GB memory, and 4 storage bays. The specifications of new file server are summarized in the Table 1.

Table 1: Specifications of the New File Server

Model	QNAP TS-431P2-4G
OS	QTS 4.3.4
CPU	1.7 GHz Quad Core (Arm Cortex-A15)
Memory	4 GB (DDR3) × 1
Storage	3 TB × 4 (RAID6)
LAN	GbE × 2 (Link Aggregation)

We continue to use previous file server as read-only storage for some time and evaluate the durability and reliability of the new file server.

INTRODUCTION OF EPICS ARCHIVING SYSTEM

Until now, we used homemade shell scripts to preserve various parameters. The timing of preservation is left to the judgment of the operator. Therefore, it was not possible to save parameters during non-operation.

In order to deal with this problem, we started experimental use of the EPICS Archiver Appliance [6] from December 2017. Approximately 140 PVs were recorded at the shortest period of 0.1 second. The ArchiveViewer [7] is used to display the acquired data. Figure 3 shows vacuum data taken from January 1, 2018 to February 1, 2018.

Figure 3: Vacuum Data from 1 Jan. 2018 to 1 Feb. 2018 taken by "The EPICS Archiver Appliance". The Viewer is "ArchiveViewer".

Since the EPICS Archiver Appliance was stable over 3 months, we started to use full-scale usage from April 2018. We prepared exclusive PC and EPICS archive server had started to record about 310 PVs. The hardware specification of the EPICS archive server are summarized in the Table 2 and software specifications are summarized in the Table 3.

Table 2: Hardware Specifications of EPICS Archive Server

CPU	4 GHz Quad Core (i7 6700K) × 1
Memory	16 GB (DDR4 PC4-17000) × 4
Storage	1 TB SSD × 1
	6 TB NAS (RAID6)

Table 3: Software Version of EPICS Archive Server

OS	CentOS7
Archiver Appliance	22 June 2017
Java	Oracle Java SE Dev. Kit 8
Tomcat	7.0.86
Apache	2.4

The data acquired by this archiver is directly stored in the NAS. Currently, the event rate is 65/s and the data consumption is 0.1 GB/Day. Since there is sufficient margin in performance, we plan to increase the number of PVs to store.

MIGRATION OF GPIB CONTROL ENVIRONMENT

In the KURNS control system, GPIB is used to control old type power supply and stepping motor controller. Control programs for the GPIB devices were made using the Glade [8] version 2 for GUI creation under the CentOS5 environment.

As the CentOS5 support period expired in March 2017, the immigration of the GPIB control environment became necessary.

In the KUNRS control system, GPIB-USB-HS [9] manufactured by National Instruments has been used as a standard device for connection between the device controlled by the GPIB and the PC. But the device driver of GPIB-USB-HS provided by manufactured is only compatible with CentOS5 or older.

In order to use the CentOS6 or later version, it was necessary to change the device driver. Therefore, we decided to use the device driver Linux GPIB Package [10] which is being developed as open source. In addition, since this device driver was able to cooperate with Python, we decided to use Tkinter as a GUI creation tools, which is Python's standard library.

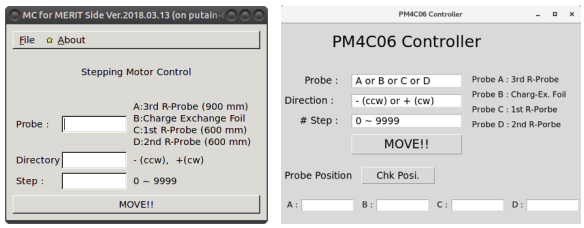
Before the migration, when relocating programs between PCs, there was a need to recompile. After migration, recompile doesn't need and portability improved with Python. Table 4 shows the results before and after the immigration of the GPIB control environment.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP13

Table 4: Migration of GPIB Control Environment

	Before Migration	After Migration
Hardware	NI GPIB-USB-HS	NI GPIB-USB-HS NI GPIB-USB-HS+
OS	CentOS5	CentOS7
Dev. Driver	NI-488.2	Linux GPIB Package
App. Lang.	C (GTK)	Python (Tkinter)

Figure 4 shows two operator interfaces of stepping motor controller, one is created using GTK and another is using Python(Tkinter).



(a) GTK based OPI. (b) Python (Tkinter) based OPI.

Figure 4: Examples of Operator Interface for the device controlled via GPIB.

NEXT PLANS

Integration of GPIB Control to EPICS

GPIB controlled devices are independent control, not in cooperation with EPICS. For this reason, different operation systems are mixed in control, which is a cause of complicating the operation of the accelerator. In order to solve this problem, we are preparing a bridge program including GPIB control which behaves as EPICS IOC.

Integration of Beam Diagnosis to EPICS

Since problems arise in the accessibility of data acquired by the beam diagnosis device, we would like to implement bulk management using EPICS archive server. We are developing data acquisition system integrated with EPICS.

Update of Linac Control System

We introduced a linear accelerator to the KURNS complex system in 2008. It has been over 10 years since the introduction, and in 2015, we had experienced serious hardware problems with RFQ.

The PC used for control has not been updated since its introduction and failed at startup and sudden shutdown occurred. Although early replacement is necessary, there is no source code of control software in the manufacture. And there is no device driver of the input/output board (for communication with the linear accelerator power supply unit) for the latest OS.

Since there are many uncertain factors regarding the updating work of the control PC, the manufacture cannot estimate the cost of renewal. However, for the continuous operation of the KURNS complex, the replacement of linac control system cannot be avoid.

SUMMARY

The KURNS complex has also exceeded 10 years since the start of construction. During this 10 years, the structure of the accelerator continues to change, and the control system is also changing to cope with the change.

For the purpose of the sophistication and stability, replacement of LabVIEW-based control system to EPICS is almost completed and for the data protection, file servers were replaced fourth times and EPICS archive server is introduced. And beam diagnosis system and GPIB control system are also moving to EPICS environment.

At present, the update plan of the control system of the linear accelerator has not been determined but this is a most urgent task of control group.

REFERENCES

[1] T. Uesugi *et al.*, "FFAGs for the ERIT and ADS Projects at KURRI", in *Proc. EPAC'08*, Geneva, Switzerland, Jun. 2008, paper TUOBM04.

[2] C. H. Pyeon *et al.*, "First Injection of Spallation Neutrons Generated by High-Energy Protons into the Kyoto University Critical Assembly", *J. Nucl. Scie. Technol.*, vol. 46, no. 12, pp. 1091–1093, 2010. doi:10.1080/18811248.2009.9711620

[3] M. Tanigaki *et al.*, "Control system for the FFAG complex at KURRI", *NIM*. A, vol. 612, pp. 354-359, 2010. doi:10.1016/j.nima.2009.11.024

[4] Y. Kuriyama *et al.*, "EPICS Control System for the FFAG Complex at KURRI", in *Proc. ICALEPCS'13*, San Francisco, CA, USA, Oct. 2013, paper THPPC036.

[5] QNAP, https://www.qnap.com

[6] The EPICS Archiver Appliance, https://slacmshankar.github.io/epicsarchiver_docs/index.html

[7] The ArchiveViewer, https://slacmshankar.github.io/epicsarchiver_docs/archiveviewer.html

[8] Glade, https://glade.gnome.org

[9] GPIB-USB-HS, http://www.ni.com/en-us/support/model.gpib-usb-hs.html

[10] Linux GPIB Package, https://linux-gpib.sourceforge.io

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP14

VACUUM CONTROL SYSTEM FOR THE TAIWAN PHOTON SOURCE

Y-C. Yang, J-Y. Chuang, Y-M. Hsiao, Y. Z. Lin, C. K. Chan and C. C. Chang
National Synchrotron Radiation Research Center (NSRRC), Hsinchu 30076, Taiwan

Abstract

The Taiwan Photon Source (TPS) is a 3 GeV storage ring. A NI C-RIO controller, basic to the real-time, EPICS program, is used and designed such as to maintain ultra-high vacuum conditions and protect vacuum components. Pressure readings from ionization gauges are taken as the logic signal to control sector gate valves to protect ultra-high vacuum condition. Monitoring of vacuum components, water-cooling systems and chamber temperatures serves to protect vacuum equipment from radiation power. The evolution and status of the control system is presented in this paper.

INTRODUCTION

Commissioning for the TPS, a low-emittance 3-GeV synchrotron ring, started in December 2014 and is now currently operated in top-up mode at 400mA for users. During past year's operation, the design goal of 500mA beam current was archived on December 2015. Until the last machine shut down in June 2018, a total beam dose of 3631 Ah was accumulated and the beam cleaning effect decreased to 1.43×10^{-11} Pa/mA. Table 1 lists some operational milestones of the TPS in past years.

Table 1: Milestones for the TPS Currently in Operation

Date	Milestone
2014.12	first beam stored
2015.03	100mA beam current archived
2015.12	500mA design goal archived
2016.09	open to user (300mA)
2016.11	1000 Ah beam dose accumulated
2017.11	400mA operational Beam Current

The TPS storage and booster rings are located in the same tunnel. The storage ring with a circumference of 518.4m, is divided into 24 sections, including 24 bending and 24 straight sections. Each section corresponds to one control instrument area (CIA). Figure 1 shows a schematic 3D drawing of the TPS vacuum system with the storage ring (SR), booster ring (BR) and control-instrument area (CIA). The vacuum component connection cable lengths between tunnel and CIA are 20 to 30 meter long.

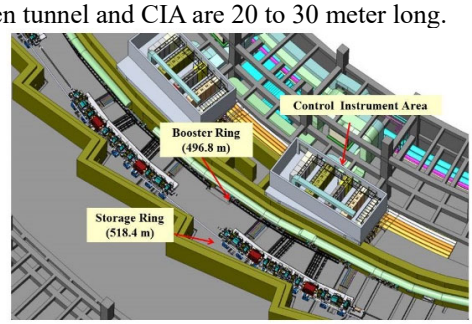


Figure 1: Schematic 3D drawing of the TPS vacuum system.

Figure 2 illustrates the layout for 1/24 of the TPS vacuum system, consisting each of a straight and a bending section with two sector gate valves (SGV), two pumping gate valves (PGV), two front-end valves (FEV), six metal angle valves (MGV), six ionization gauges (IG), ten non-evaporable getter (NEG) pumps, six sputtering ion pumps (IP), and eight turbo-molecular pumps (TMP). The vacuum chambers inside the cell contain two straight ducts S3, S4, and two bending chambers B1, B2; the S1 and S2 ducts are located at both ends of the cell isolated with two SGVs [1].

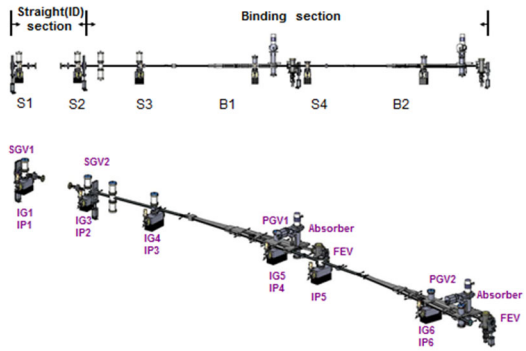


Figure 2: Layout of 1/24 vacuum section.

The mechanism of the vacuum control system is to maintain and protect ultra-high vacuum conditions by controlling and monitoring vacuum components as described above. The safety interlock system is based on the conditions in vacuum components, such as a gauge, ion pump or valve. The following sections describe the design concepts.

VACUUM CONTROL SYSTEM

TPS uses EPICS (Experimental Physics and Industrial Control System) to control and monitor the accelerator machine. EPICS can provide a standard client-server model for a distributed system. In the TPS vacuum control system, a Compact-RIO real-time controller from National Instrument® serves for the vacuum safety interlock, data acquisition and monitoring systems. Between the C-RIO and vacuum system, the interface of I/O communication is used by the I/O connect port of the vacuum controllers , such as vacuum gauges, pumps and meters for cooling water, or directly by the I/O terminals of the vacuum components. The architecture of the control and communications relations is shown in Fig. 3.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP14

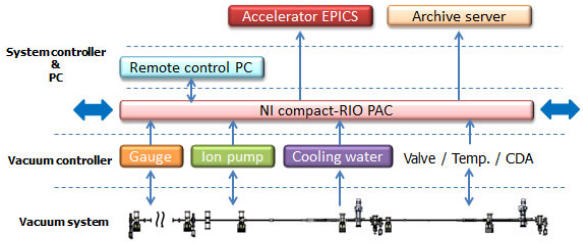


Figure 3: Architecture of the vacuum control and communications system.

Since the vacuum system of the storage ring is divided into 24 sections, with 24 C-RIO controllers distributed into 24 CIA associated with 24 vacuum sections. Each C-RIO collects all signals from one section, including about 48 analogue input signals, 96 digital input signals and 64 digital output signals. Among the analogue signals, the vacuum gauge and pump pressure readings are taken as the basic logic signal for the safety interlock system. The RTD temperature sensor readings serve to monitor the cooling water and vacuum components without special cooling system such as valves, bellows and BPM blocks. Digital input and output signals provide the status , set-point, logic trigger and remote control of the vacuum components [2]. A model of the 9074 C-RIO controller with three analogue input modules, three digital input and two digital output modules installed in CIA is shown in Fig. 4.

Figure 4: A model of the C-RIO controller with modules.

In addition to the storage ring, vacuum signals of the booster ring (BR), the transfer lines between linac and booster ring (LTB) and between booster ring and storage ring (BTS) are connected to adjacent C-RIO controllers, depending on the location, for safety interlock, data acquisition and monitors. Figure 5 shows the number of I/O ports for each vacuum subsystem.

Sub-system	AI		DI	DO
	reading	RTD	status / trigger	control
LTB	3	N/A	22	13
BR	108	N/A	264	192
BTS	6	N/A	32	27
SR	144	360	1152	1008
UTILITY	168	168	480	N/A

Figure 5: I/O ports for the vacuum subsystem.

SAFETY INTERLOCK SYSTEM

The protection of the ultra-high vacuum condition and of vacuum components is the main concern for the safety interlock system. Before the vacuum control system is designed, the properties of the vacuum components must be considered, especially the safety interlock system. Some considerations follow.

(1) Vacuum-gauge protection: In the storage ring of the TPS, an ionization gauge was chosen and taken as the source for a logical signal. During machine operation, the readings of a vacuum gauge may increase to more than 10^{-6} Pa. To avoid vacuum gauges operating in conditions of poor vacuum for an extended period, which would decrease their lifetime, a self-protection mode is set. In this mode, vacuum gauges become automatically switched off when the vacuum pressure increases suddenly to more than 1×10^{-3} Pa, but can be switched on only manually after the pressure recovers.

(2) Ion-pump protection: Similar to the vacuum gauges, ion pumps are switched on only when the local pressure is less than 1×10^{-4} Pa according to the logical output signal of the vacuum gauges. A protection mode of an ion-pump controller is concurrently selected. In this mode, the controller limits the output current and switches off the high voltage when the output current reaches or exceeds a threshold current by more than 0.2 s.

(3) Isolation valve: The mechanism of the safety interlock system is set to control the opening and closing of the sector gate valves (SGV) to isolate a vacuum system with poor pressure. When the pressure increases to more than 1×10^{-4} Pa, which is the trigger output of the vacuum gauges at either end of the valve, the SGV closes to protect the vacuum at the other side. Two properties of the SGV must be considered here: the pressure of compressed air and the closing time. The pressure of compressed air for normal operation is 4~8 bar (4.08~8.16 kg/cm^2). A trigger point of 5 kg/cm^2 for the compressed air is therefore set as the interlock trigger signal for the utility system. The closing interval of the SGV is 4 s, based on sufficient pressure and rate of flow of compressed air to fill the cylinder. To ensure normal operation of the SGVs, independent air piping for each SGV is necessary. If one air pipe supplies more than one SGV, the closing time of the SGVs is delayed, thus affecting the performance of the SGVs.

LOGIC DESIGN

Figure 6 shows the logic diagram of the SGV control mechanism, which complies with the principle of manual on and fail-safety. When the pressure on both sides of an SGV is satisfactory, the SGV can be controlled and switched off automatically as soon as the interlock at either end is triggered. All three gauges are installed between two SGVs. If any two gauges are over the set point or

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP14

malfunctioning, the logic trigger becomes active and the SGVs are then closed. Two front- ends (FE) associated with this vacuum system additionally ensure the completeness of the safety interlock system. Besides, considering the vacuum pressure, the emergency trip for neighbouring valves is added to the interlock system to decrease the risk of a spread of poor vacuum. When any neighbouring valve is out of control or malfunctions, the emergency trip signal becomes active and the SGVs close to prevent the spread of poor vacuum.

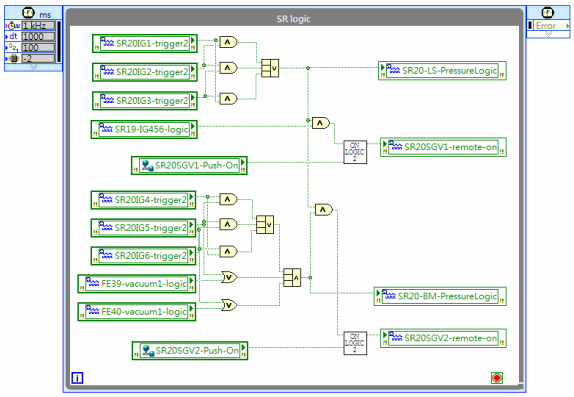


Figure 6: Diagram of the SGV logic control system.

Besides the basic protection of ultra-high vacuum condition, another issue is about vacuum component protection from synchrotron radiation power. The open signal of an SGV and normal status of the cooling water are summed and sent to the machine protection system (MPS) to make sure the vacuum system is ready for machine operation.

During the past years of operation, the interlock logic was optimized. The biggest difference was to avoid a machine trip by a single signal source. Several times during machine operation, false FE vacuum signals caused interrupts from faulty evaluation of signals from FEs and BLs. A vacuum signal in the storage ring adjacent to the FE system was added and a trip signal is given only when these two signals fault simultaneously. The temperature interlock system was also corrected. Countdown and reset functions were added to avoid a trip signal being sent by the signal from a damaged temperature sensor [3].

MONITOR SYSTEM

NI Labview is a system engineering software, which offers a graphical programming approach that visualizes the application. The graphical interface is flexible and easy to use. Web publishing functions in Labview are widely used in the TPS vacuum monitoring system, which transfer Labview front panel pages to web pages easily. Figure 7 displays one example for the monitoring page of the vacuum status, which includes machine operation conditions and insertion devices status acessed from the TPS EPICS IO server and vacuum pressure signal binding from 24 C-RIO controllers. It is easy to be viewed by vacuum group staff on PC or cell phone.

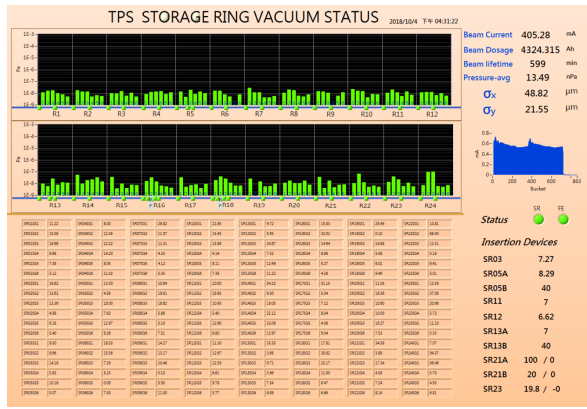


Figure 7: Vacuum monitoring page of the TPS storage ring.

One fake event occurred originating from the Front End (FE) interlock system, which also uses a C-RIO 9074 controller [4]. The reason for the event was not clear, but experience indicats that it was easy occured after long time operation without rebooting or execution of the distributed system manager (DSM) function of Labview. The CRIO 9074 VxWorks OS controllers were replaced to Linux OS versions, since the C-RIO 903X Linux OS controller exhibits a more stable performance, the CPU loading is five times lower than for the 9074 VxWorks.

CONCLUSIONS

The design of the TPS vacuum control system is described above. The vacuum pressure protection function and component protection logics worked well during the past years of operation. All vacuum signals, including vacuum pressure, status of vacuum devices, temperature distribution were displayed on web pages which is easy to monitor. More applications like LINE Notify instant message sending function will be developed in the future.

REFERENCES

[1] G. Y. Hsiung, *et al.*, "TPS Vacuum system", in *Proc. PAC'09*, Vancouver, Canada, May 4-8, 2009, paper MO6RFP018.

[2] Y. C. Yang, *et al.*, "Development of the TPS Vacuum Interlock and Monitor System", in *Proc. IPAC`14*, Dresden, Germany, June 15-20, 2014, paper WEPME051, pp. 2387-2389.

[3] Y. C. Yang, *et al.*, "Two year operational experience with the TPS vacuum system", Journal of physics: conf. Series 874(2017).

[4] J. C. Chuang, *et al.*, "Upgrade for the TPS Fail Safe Front End Interlock System", 9th International Workshop on Radiation Safety at Synchrotron Radiation Sources, Hsinchu, Taiwan, April 19-22, 2017.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP15

RECENT DEVELOPMENT OF THE RIKEN RI BEAM FACTORY CONTROL SYSTEM

M. Komiyama, M. Fujimaki, N. Fukunishi, K. Kumagai, A. Uchiyama
RIKEN Nishina Center, Wako, Saitama, Japan

Abstract

We report on the development of the successor to the existing controller devices used for the magnet power supplies in the RIKEN Radioactive Isotope Beam Factory (RIBF). The existing system controlling the magnet power supplies is operated on the Versa Module Europa (VME) computing machines, under the Experimental Physics and Industrial Control System (EPICS) framework. The present controller system has been operated stably for over 10 years. However, it is now commercially unavailable, because the supply of some parts has already ceased. From 2011 to 2016, we have been developing a successor system to achieve essentially the same function as the existing one, but the successor system is designed to run in control systems constructed by programmable logic controller (PLC) modules instead of the VME computing environment, in order to achieve a cost reduction and easily cooperate with other systems.

We set up a test system using this successor and confirmed that a magnet power supply could be controlled in the same manner as the existing system. Now, we plan to begin controlling magnets of beam transport lines using this successor system in the current year.

INTRODUCTION

The RIKEN Radioactive Isotope Beam Factory (RIBF) is a cyclotron-based accelerator facility aiming at the development of nuclear physics, materials, and life science studies. RIBF consists of two heavy-ion linear accelerator injectors and five heavy-ion cyclotrons, including the world's first superconducting ring cyclotron (SRC). Cascades of the cyclotrons can provide the world's most intense RI beams over the whole atomic mass range, using fragmentation or fission of high-energy heavy-ion beams [1]. For example, a 345-MeV/nucleon ^{238}U beam of 70 pnA was successfully extracted from the SRC in 2017. RIBF was constructed as an extension of the old facility commissioned in 1986, by adding three new cyclotrons, and began operation in 2006.

The components of the RIBF accelerator complex, such as the magnet power supplies, beam diagnostic devices, and vacuum systems, are controlled by the Experimental Physics and Industrial Control System (EPICS) [2], with a few exceptions such as the control system dedicated to the radio frequency system of RIBF [3]. However, all the essential operation datasets of EPICS and other control systems are integrated into the EPICS-based control system. In addition, two types of interlock systems that are independent of the accelerator control systems are also operated in the RIBF facility: a radiation safety interlock system for human protection [4] and a beam interlock system (BIS) that protects the hardware of the RIBF accelerator complex from potential damage caused by high-power heavy-ion beams [5].

UPDATE OF CONTROLLER DEVICE FOR MAGNET POWER SUPPLY

Controller Device for Magnet Power Supplies in RIBF

The magnet power supplies are operated both in the old facility section and in the newly added facility section commissioned since 2006 (hereafter, the new facility section). However, there are differences in these controllers according to their introduction times. The magnet power supplies in the old facility section are controlled by our in-house controller device based on Computer-Aided Measurement And Control (CAMAC), a device interface module (DIM) [6]. On the other hand, the magnet power supplies in the new facility section are designed to be controlled by the Network I/O (NIO) system, which is a commercially available control system manufactured by the Hitachi Zosen Corporation. A block diagram of the NIO system is presented in Fig. 1.

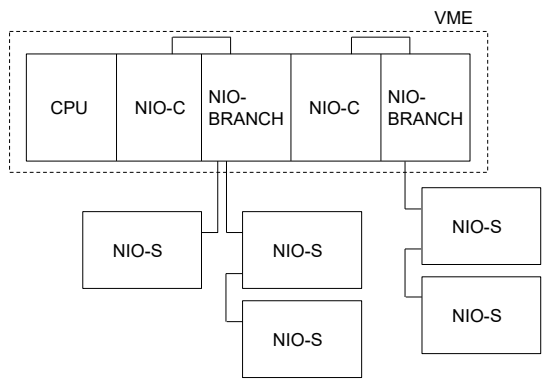


Figure 1: Block diagram of the NIO system under operation.

The NIO system consists of three types of controllers: the NIO-S board, NIO-C board, and branch board. The NIO-S board is a slave board attached directly to the magnet power supply, which controls it according to a signal sent from an upper-level control system through the NIO-C board. The NIO-C board acts as a master board of the NIO-S boards and is designed to operate in the Versa Module Europa (VME) computing machines. The NIO-C board not only has an NIO board, but also an additional board dedicated to High level Data Link

* misaki@riken.jp

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP15

Control procedure (HDLC) communication. The NIO-C and NIO-S boards are connected using an optical fiber cable through a branch board. One NIO-C board can control 43 NIO-S boards via branch boards, and there are 15 NIO-C boards in operation connected to seven VME chassis distributed in RIBF. Because one NIO-S board can only control one magnet power supply, there are approximately 500 NIO-S boards in the RIBF accelerator complex, and this corresponds to 60% of the total magnet power supplies used in the RIBF accelerator complex.

Development and Update to the Successor

DIM has been working stably for over 30 years, with improvements being added as required. However, in addition to its serious aging, it has become difficult to maintain DIM because there are no engineers who can produce or repair it. Considering this situation, we are systematically replacing the old magnet power supplies controlled by DIM to new ones controlled by the NIO. As a result, the number of magnet power supplies controlled by DIM was reduced from over 680 to 375 in 2017. In this situation, the production of NIO was terminated, and we can no longer purchase NIO boards. Therefore, we needed to develop a successor, and we planned to develop this in the order of an NIO-S board, NIO-C board, and branch board. We developed an NIO-S successor between 2011 and 2014. Table 1 lists the hardware specifications of the NIO-S under operation and its newly developed successor.

Table 1: Specifications of NIO-S Under Operation and its Successor

NIO-S	Under operation	Successor
CPU	SH-2 28.33 MHz	SH-2 SH7084 32 MHz
EPROM	128 Kbyte	-
Flash ROM	256 Kbyte	32 Mbyte
SRAM	512 Kbyte	512 Kbyte
SCA	HD64570	TD-HDLCip
Digital input (points)	32 (8 points/ 1 common)	48 (8 points/ 1 common)
Digital output (points)	32 (8 points/ 1 common)	32 (8 points/ 1 common)
Serial interface	RS-485 Optical link	RS-485 Optical link
Debug port	RS-232C	RS-232C

This successor was designed to be compatible with the NIO-S board currently under operation. Communication is the most significant feature of the successor. The HDLC transmission method, which is performed using an adaptor chip used for serial communication in the NIO-S under operation, is replaced in the successor by using the IP in the field-programmable gate array (FPGA). We also increased the number of digital inputs from 32 to 48, in consideration of future expandability.

After the development, we updated the existing NIO-S installed in the three independent magnet power supplies to the successor and began testing its operation from 2014. We can control these as smoothly as other magnet power supplies without modifying anything in the existing control programs.

In the development of a successor for the NIO-C board, the required specifications are essentially the same as for the existing board. However, we decided to design a successor to run on a control system constructed by programmable logic controller (PLC) modules instead of the VME computing environment used currently, in order to achieve a cost reduction and easily cooperate with other systems. The hardware specifications of the NIO-C under operation and its newly developed successor are listed in Table 2.

Table 2: Specifications of the NIO-C Under Operation and its Successor

NIO-C	Under operation		Successor
	NIO-board	Communication-board (HDLC)	
CPU	TMPZ84 C013A-10 6.4 MHz	TMPZ84C0 13A-10 8 MHz	NIOS2 Processor 32 MHz
ROM	28 Kbyte	28 Kbyte	EPCS16 2 Mbyte
RAM	32 Kbyte	28 Kbyte	
DRAM	-	-	32 Mbyte
DPRAM	2 Kbyte	2 Kbyte	16 Kbyte (in FPGA)
SCA	HD64570		TD-HDLCip
FPGA	-		Cyclone4 EP4CE22
Bus controller	- (VME)		A6374LG (Yokogawa)
Serial interface	RS-485		RS-485
Debug port	RS-232C		RS-232C

The NIO-C successor was developed based on the PLC system manufactured by the Yokogawa Electric Corporation (hereafter, FA-M3), following recent trends in the control systems of RIBF accelerators. One of the advantages of adopting FA-M3 is that we can set up a

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP15

simple control system by choosing a Linux-based PLC-CPU (F3RP61 [7]) on which EPICS programs are executed. In that case, F3RP61 works not only as a device controller, but also as an EPICS Input Output controller (IOC) [8]. Figure 2 presents a block diagram of the successor system of the NIO system. We designed the NIO-C successor to communicate data and commands with a PLC-CPU via a shared RAM contained on it.

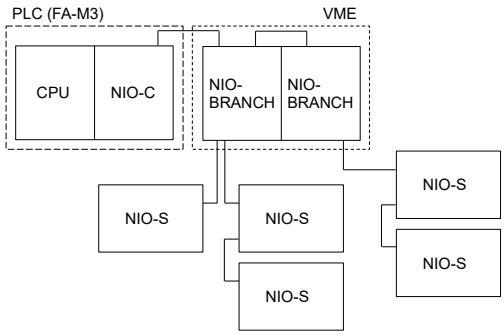


Figure 2: Block diagram of the successor system of the NIO system.

The software of the successor was developed maintaining full compatibility with the NIO-C under operation. The NIO-C successor has 13 kinds of command codes, such as magnet power supply on/off, polarity switching, and pulse output. Most of these command codes inherit those of the NIO-C under operation: there is one updated command code for setting current value to the magnet power supply. In the existing system, there are two ways of setting a current target value for a magnet power supply: one is to give only the target current value to the magnet power supply, in which a default current excitation rate is used; the other is to provide a target current value and specify its current excitation rate (hereafter ramp control). The ramp control command code in the existing system requires setting parameters such as the current target value, step time, and total setting time to reach the target current value. It is assumed that the current value is 0 A, and ramp control is performed from 0 A towards the target current value. In other words, this cannot be used when starting from other current values. Therefore, we now only use former command code in the current control of the magnet power supply. When we need to significantly change the current value, we employ a program developed by implementing ramp control starting from an arbitrary current value, in order to ensure safety and not damage the magnet power supply. Therefore, in developing the software for the NIO-C successor an upgraded command code for ramp control was developed by adding a current starting value to the parameter set with the existing ramp command. We assume that this command code is utilized in the control of a large current magnet power supply, such as for a dipole magnet.

We completed the development of the software of the NIO-C successor in 2016. Subsequently, we have started to develop the control program of the magnet power supply, using F3RP61 as an EPICS IOC. As a first step, we have developed a program that implements equivalent usability to existing control programs by using a test unit containing only F3RP61 and the NIO-C successor, as shown in Fig. 2. The test unit has been attached to one of the NIO stations in operation, which controls the magnets of beam transport lines including dipole magnets, quadrupole magnets, and steering magnets. There is one VME-CPU, five NIO-Cs, and two or three branch boards connected to each NIO-C, and approximately 150 of the magnet power supplies are controlled by this NIO station. For testing the NIO-C successor, we replaced one of the NIO-Cs in operation with the successor and connected this to an existing branch board. This NIO-C is controlling 34 magnet power supplies. As a first step, we tested all 13 command codes by developing test programs, and we confirmed that all commands successfully performed according to their specifications. Based on this result, we have developed an EPICS run-time database and graphical user interface (GUI) for the magnet control based on the existing one. As a result, we successfully controlled all 34 magnet power supplies in the same manner as the existing system. The response to the command input was the same as for the current system. Furthermore, we can control these by using an updated command for the ramp control by setting the parameters of some patterns.

As a next step, we will start the development of a successor to the branch board. The branch board is a non-intelligent board for a star connection between an NIO-C and NIO-S with an optical cable. Although the existing branch board runs in the VME chassis, a necessary input power of 5 V should be fed independently from VME. Therefore, we plan to develop a successor to the branch board with specifications of not using VME, in order to lower the costs and simplify the board.

REFERENCES

[1] O. Kamigaito *et al.*,"Present Status and Future Plan of RIKEN RI Beam Factory", in *Proc. IPAC2016*, Busan, Korea, May 2016, paper TUPMR022, pp. 1281–1283. doi:10.18429/JACoW-IPAC2016-TUPMR022

[2] EPICS, http://www.aps.anl.gov/epics/

[3] M. Komiyama *et al.*, "Recent Improvements to the RIKEN RI Beam Factory Control System", in *Proc. PCaPAC2016*, Campinas, Brazil, Oct. 2016, paper WEPOPRP011 pp. 31–34. doi.org/10.18429/JACoW-PCaPAC2016-WEPOPRP011

[4] H. Sakamoto *et al.*, "HIS: The radiation safety interlock system for the RIKEN RI Beam Factory," RIKEN Accel. Prog. Rep. vol. 37, pp. 281, 2004.

[5] M. Komiyama, M. Fujimaki, I. Yokoyama, and M. Kase, "Status of control and beam interlock system for RARF and RIBF," RIKEN Accel. Prog. Rep. vol. 39, pp. 239, 2006.

[6] K. Shimizu, T. Wada, J. Fujita, and I. Yokoyama, *in Proc. Cyclotrons'84*, pp. 392-395.

[7] FA-M3, http://www.FA-M3.com/jp/

[8] A. Uchiyama *et al.*, in *Proc. PCaPAC2008*, pp.145–147.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP16

CMS ECAL DETECTOR CONTROL SYSTEM UPGRADE PLAN FOR THE CERN LARGE HADRON COLLIDER LONG SHUTDOWN II

R. Jiménez Estupiñán, D. Di Calafiori, G. Dissertori, L. Djambazov, W. Lustermann, S. Zelepukin[1], ETH, Zurich, Switzerland
[1]also at University of Wisconsin-Madison, USA

Abstract

The Electromagnetic Calorimeter (ECAL) is one of the detectors of the Compact Muon Solenoid (CMS) experiment at the CERN Large Hadron Collider (LHC). The ECAL Detector Control System (DCS) software has been implemented using the WinCC Open Architecture (OA) [1] platform. Modifications that require fundamental changes in the architecture are deployed only during the LHC long shutdowns. The upcoming long shutdown (2019-2020) offers a unique opportunity to perform large software updates to achieve a higher modularity, enabling a faster adaptation to changes in the experiment environment. We present the main activities of the ECAL DCS upgrade plan, covering aspects such as the re-organization of the computing infrastructure, the consolidation of integration tools using virtualized environments and the further usage of centralized resources. CMS software toolkits are evaluated from the point of view of the standardization of important parts of the system, such as the machine protection mechanism and graphical user interfaces. Many of the presented features are currently being developed, serving as precursors to the major ECAL upgrade foreseen for the next long shutdown (~2024-2025).

INTRODUCTION

The CMS Electromagnetic Calorimeter (ECAL) detector is composed of a scintillating crystal calorimeter and a lead/silicon preshower. The CMS experiment takes collisions data at the LHC, requiring extremely high reliability and the minimum down-time of the various detectors. The ECAL detector is subdivided in partitions as follows: Barrel (EB), Endcaps (EE) and Preshower (ES). The EB partition consists of 36 Supermodules (SM) each containing 1700 crystals, the EE consists of two endcaps split in four semi-circles (Dees), each containing 3662 crystals, and the ES consists of two circular structures.

The DCS controls and monitors the status of the hardware of each partition: cooling, powering systems, safety systems, environmental monitoring systems and other external interfaces. The DCS software coordinates the interaction between the different subsystems, providing an effective and meaningful way of operating the detector. From the architectural point of view, the DCS software is built by a hierarchy of components, whose main features can be organized into one of the following categories: Systems Configuration (peripheral addresses, database archivers, alarms, notifications, etc.), Finite State Machine (FSM), Automatic Actions (AA) and User Interfaces (UIs).

The CMS ECAL DCS [2] has successfully supported the ECAL operations since the CMS commissioning, more than 10 years ago. The CMS detectors maintenance is mainly driven by the LHC schedule, often restricted to periods of 2 to 5 days, called Technical Stops (TS). Major modifications are postponed to the Extended Year-end Technical Stops (EYETS) or the LHC Long Shutdowns (LS). The next LS will last for about two years, starting at the end of 2018. During this period, also known as the Second Long Shutdown (LS2), multiple activities across the LHC, Injectors and LHC Experiments will be performed [3]. Following the previous re-integration and consolidation of the CMS ECAL DCS [4], the LS2 provides an ideal opportunity to bring new features into operation, improve the maintainability of the CMS ECAL DCS software and increase the level of availability of the systems.

DCS UPGRADE PLAN OVERVIEW

Multiple activities are scheduled for the LS2; however, this paper focuses on those that imply significant changes to the DCS architecture:

- Re-configuration of the low voltage powering system. The device distribution at the Controller Area Network (CAN) level will be modified to increase the overall performance of the system;
- Extension of the protective automatic actions. The plan includes a proposal to increase the granularity of the current actions, the implementation of new ones and the evaluation of existing frameworks;
- Software standardization. The existing development guidelines will be extended and applied during the creation/modification of every component.
- Consolidation of the CMS ECAL DCS test platform. The usage of CMS automated deployment and testing tools will contribute to have faster development cycles, while reducing the time spend on maintenance;
- Improvements in the Programmable Logic Controller (PLC) based protection and safety systems.

Some of the LS2 activities present dependencies that define the order in which they must be executed (e.g. software updates come after the changes in the hardware layout). The LS2 upgrade plan is organized according to the tasks priority and dependencies (See Fig. 1) and it will be refined in the upcoming months, in order to meet all the requirements presented by the ECAL community.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP16

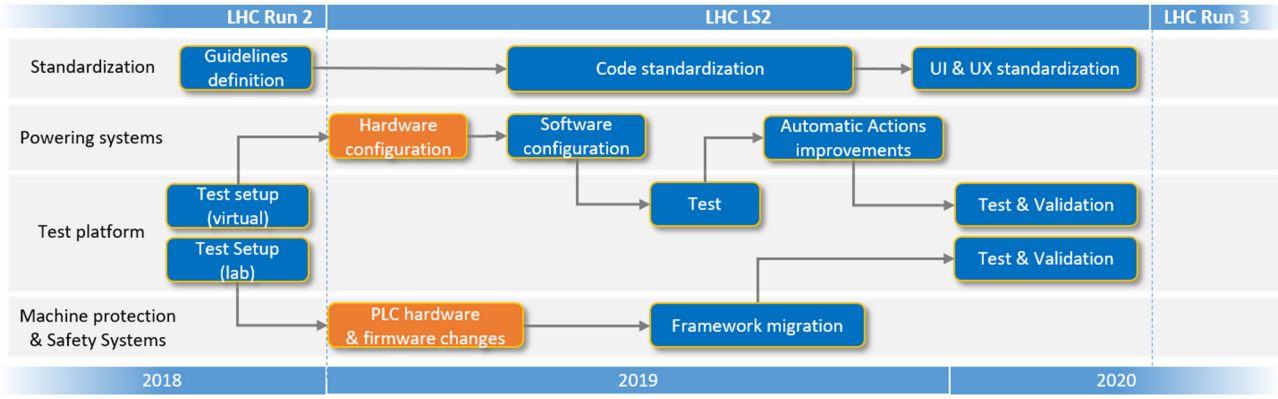


Figure 1: ECAL DCS upgrade plan for the LS2.

RECONFIGURATION OF THE LOW VOLTAGE POWERING SYSTEM

The ECAL powering systems are classified into two categories: Low Voltage (LV) and Bias Voltage (BV) power systems. The LV system for the EB and EE partitions is composed of 136 Wiener [5] Marathon power supplies, distributed across 10 CAN buses. The CAN buses for the LV system are connected to CAN-to-Ethernet interfaces enabling the access over the CMS private Ethernet network. The EB/EE BV and ES LV and BV are based on CAEN [6] power supplies using built-in Ethernet interfaces.

In the past years, the connection between the DCS and the EB/EE LV system has been affected by disruptions in the CMS private Ethernet network. When the communication is disrupted, the driver qualifies the data as invalid, similarly to an internal failure of the power supplies. Upon this condition, the CMS ECAL DCS is programmed to shut down the affected partitions (SMs or Dees) as a preventive action. After a few of such incidents, several tests were performed to understand the data invalidation process with the help of a network disruptor device. The tests focused on the two aspects: the disruption time required by the driver to invalidate the data and the time required to reconnect and restore the communication. Among other things, the tests revealed that the performance of the driver depends on the distribution of devices across the different buses. This means that the most populated bus determines the overall recovery time after such incident. This particular conclusion motivated the following activities for the LS2: extension of the number of buses connected to the CAN bus interface from 10 to 12 buses, redistribution of devices across the buses (maximum 15 devices per bus) and the reorganization of the DCS software according to the new interface layout (See Fig 2.). These changes, together with a new driver's configuration, will help to increase the overall performance of the LV network, making the DCS more tolerant to disruptions. This task requires the assignment of new peripheral addresses, the renaming of multiple structures and a crosswise reconfiguration of the software to ensure the correct integration of the changes.

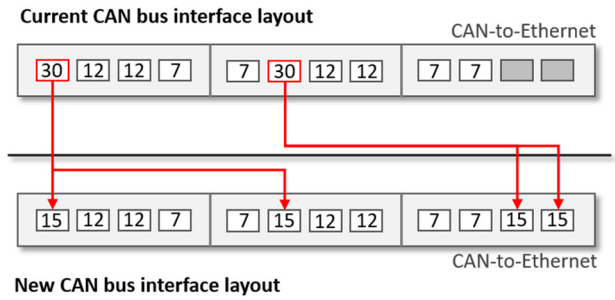


Figure 2: Changes in the CAN-to-Ethernet interface layout. The number of power supplies in each CAN bus is indicated in the white squares.

EXTENSION OF THE PROTECTIVE AUTOMATIC ACTIONS

The automatic actions in the DCS are preventive measures intended to protect the detector before escalating the problem to the safety systems. Actions are programmed to prevent delivering power during harmful conditions (e.g. absence of cooling, high humidity, etc.). The current implementation will be analysed and possibly extended to include the following features:

- Creation of new triggering conditions based on temperature information of the power supplies.
- Finer granularity of the automatic actions for the EE and EB partitions.
- Standardization of the protective automatic actions.

Finer Granularity of the Automatic Actions

Currently, the automatic actions for the EB and EE partitions are executed with the granularity of one SM or Dee. Each SM is powered by 3 LV Marathon power supplies and 34 BV channels. Dees are powered by 7 LV Marathon power supplies and 12 BV channels. This means that a single channel failure in a LV power supply triggers an action over a large number of devices regardless of their status, compromising the CMS data quality. A concrete proposal to increase the granularity of the automatic actions will be presented in the upcoming months, following a thorough analysis to prove the feasibility of this modification.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP16

Standardization of the Protective Automatic Actions

The protective automatic actions in the CMS ECAL DCS are programmed following different approaches [7]. The ECAL Finite State Machine (FSM) embeds most of the automatic actions for the EB and EE partitions, while the rest of the actions are implemented in custom control scripts. Other mechanisms to program automatic actions are also available to the CMS community. The Detector Protection is a software framework using the WinCC OA distribution capabilities and its native locking mechanism for exclusive data access. This framework offers a single interface to configure and monitor actions across distributed systems in a uniformed way. This framework is currently used by multiple projects within CMS and its usage will be considered after evaluating aspects such as the scalability of the software.

CONSOLIDATION OF THE TEST PLATFORM

The current CMS ECAL DCS computing infrastructure is composed of three redundant servers. Each server features 16 CPU cores, 32 gigabytes of RAM and runs the Windows 2008 Enterprise R2 operating system. The DCS software components are distributed among these servers according to their functionality and the hardware connected to them. The computing load and distribution of concerns is considered optimal, resulting in a highly cohesive system with very low coupling between the different software components. The configuration of the control platform (distributed and redundant connections, network parametrization and components layout) is stored in a central installation database.

Since the beginning of 2018, the CMS ECAL DCS team uses a set of CMS installation tools to perform unattended software deployment in the test setups. These tools are able to access the DCS installation databases and build the latest version of the software using the official code repositories. In addition to this, the installation tools are able to monitor changes and to synchronize the development copies with the production systems. The computing hardware used for the test setup is similar to the one used in production (except for the redundant set of nodes) and it will be extensively used during the LS2 for performance analysis, debugging and automated testing.

Virtual Test Environment

In addition to the physical test setup, three virtual machines from the CMS computing infrastructure are also available for testing. Each of them provides with 4 virtual CPUs, 8 GB of RAM and 40 GB of disk storage, sufficient to run an offline copy of the CMS ECAL DCS software. This virtual setup is particularly useful for developing multiple features in parallel, while minimizing the time required for maintaining the infrastructure. The usage of virtual machines permits the reinstallation of the operating system within minutes, having a ready-to-use copy of the CMS ECAL DCS software within a few hours. The virtual environment will be used during the LS2 to fragment and speed up the different software developments (See Fig 3.).

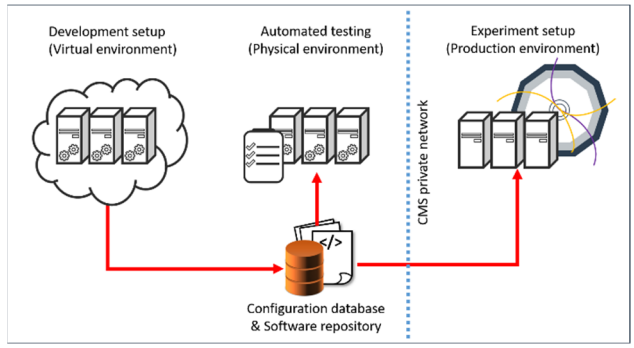


Figure 3: Software deployment process.

SOFTWARE STANDARDIZATION

The CMS ECAL DCS software was written by multiple developers, using different design criteria and programming styles. During the latest re-integration, the software was reorganized following a set of recommendations for controls systems development, in compliance with the existing JCOP framework [8] and CMS Central DCS integration guidelines [9]. Part of this reorganization work focussed on merging applications, migrating the DCS to the latest generations of software/hardware technologies, while adapting the architecture to run in fewer but more powerful servers. The resulting architecture was successfully adjusted and simplified, enabling the reduction from fifteen to three servers. Many software components execute similar tasks over different domains and areas of the detector (e.g. configuring and monitoring hardware, triggering protective automatic actions, etc.). However, the different implementations may differ substantially from each other, making the overall architecture sometimes difficult to evolve or maintain. One of the goals for the LS2 is to extend the existing guidelines, homogenize and incorporate new functionality by means of common libraries. The new guidelines will put the focus on specific naming conventions and programming best practices to improve the ECAL DCS software in the following areas:

- Code styling: Uniform naming and declaration rules for functions and variables will be introduced incrementally to improve the clarity, maintainability, extendibility of the code;
- Shared functionality: Similar tasks across components will be abstracted and encapsulated (e.g., resolution of hardware dependencies, storage of mappings, etc.)
- Debugging and diagnostics tools: New functionality will be introduced to extract run-time information from the systems and to improve the application message logging;
- User interfaces: Some of the user interfaces will be redesigned to improve information handling and to ease the support service during the on-call operations.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP16

MACHINE PROTECTION AND SAFETY SYSTEMS

The CMS ECAL detector features Siemens PLC [10] based safety systems, in addition to the CMS Safety System (DSS). The environmental conditions inside the EB and EE partitions are monitored through custom readout units, transmitting the sensors data to the PLC through RS485 buses. The environmental conditions inside the ES partitions are directly monitored by two other controllers, whose core functionality is implemented using the CMS Tracker PLC framework. The Tracker PLC framework is a compilation of tools to develop PLC applications for particle detectors. The framework permits a certain level of parametrization (alarm limits, deactivation of broken sensors, etc.) from the DCS supervisory layer. At the moment, these features are present in the two ES PLCs and some of them will be activated and integrated into the DCS interface. Further steps in the migration to this framework are also foreseen for the EB and EE Safety PLC, aiming to provide a higher level of support using the CMS ECAL DCS user interfaces.

CONCLUSION

Starting from a well consolidated software, the CMS ECAL DCS team seeks now for an efficient, extensible and more maintainable architecture. The adoption of common frameworks, development guidelines and tools will help to reduce the maintenance efforts while improving the support of critical parts of the system. In addition to this, the changes in the CAN bus interface and the protective automatic actions will contribute to increase the overall performance of the detector by providing a more reliable and robust system for the next physics operations in 2020.

ACKNOWLEDGEMENTS

The authors would like to thank the Swiss National Science Foundation for the financial support, as well as the CMS Central DCS team and the CMS Tracker DCS team for sharing some of the tools mentioned in this paper.

REFERENCES

[1] SIMATIC WinCC OA, http://www.etm.at/index_e.asp

[2] D. R. S. Di Calafiori *et al*., "The CMS ECAL Detector Control System", in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, paper THA002, p. 633.

[3] M. Bernardini *et al*., Long Shutdown 2 @ LHC, CERN, Geneva, Switzerland.

[4] O. Holme *et al*., "Re-integration and Consolidation of the Detector Control System for the Compact Muon Solenoid Electromagnetic Calorimeter", in *Proc. ICALEPCS'13*, San Francisco, USA, Oct. 2013, paper MOPPC035, p. 154.

[5] Wiener Electronics for Research & Industry, http://www.wiener-d.com/

[6] CAEN Tools for discovery, http://www.caen.it/

[7] R. Jimenez Estupinan *et al*., "Improving the safety and protective automatic actions of the CMS Electromagnetic Calorimeter Detector Control System", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, paper THPHA109, p. 1639.

[8] JCOP Framework team, "Joint Controls Project (JCOP) Framework sub-project guide-lines and conventions", CERN, Geneva, Switzerland, EDMS ID:1100577, Oct. 2010, https://indico.cern.ch/event/151538/attachments/155675/220269/jcopFrameworkGuidelines.pdf

[9] R. Gomez-Reino, CMS DCS Integration Guidelines, 2007, [online], https://twiki.cern.ch/twiki/bin/view/CMS/DCSIntegrationGuidelines

[10] Siemens PLC, https://www.siemens.com/global/en/home/products/automation/systems/industrial/plc.html

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP17

EXTENDING THE REMOTE CONTROL CAPABILITIES IN THE CMS DETECTOR CONTROL SYSTEM WITH REMOTE PROCEDURE CALL SERVICES

R. Jiménez Estupiñán, ETH Zurich, Zurich, Switzerland
Attila Racz, Christian Deldicque, Christian Wernet, Christoph Schwick, Cristina Vazquez Velez, Dainius Simelevicius [1], Diego Da Silva Gomes, Dinyar Rabady, Dominique Gigi, Emilio Meschi, Frank Glege, Frans Meijers, Hannes Sakulin, Jeroen Hegeman, Jonathan Richard Fulcher, Luciano Orsini, Maciej Gladki, Marc Dobson, Michael Lettrich, Philipp Brummer [1], Thomas Reis, CERN, Geneva, Switzerland
Ulf Behrens, DESY, Hamburg, Germany
Audrius Mecionis [2], Jean-Marc Andre, Mantas Stankevicius [1], Nicolas Doualot, Petr Zejdl[3], Remigius K. Mommsen, Srecko Morovic, Valdas Rapsevicius [1], Vivian O'Dell, FNAL, Chicago, Illinois, USA
Christoph Paus, Georgiana-Lavinia Darlea, Guillelmo Gomez-Ceballos, Zeynep Demiragli, MIT, Cambridge, Massachusetts, USA
Andrea Petrucci, Rice University, Houston, Texas, USA
Ioannis Papakrivopoulos, Technical University of Athens, Athens, Greece
Samim Erhan, UCLA, Los Angeles, California, USA
Andre Holzner, James Branson, Marco Pieri, Sergio Cittolin, UCSD, San Diego, California, USA
[1]also at Karlsruhe Institute of Technology, Karlsruhe, Germany, [2]also at Vilnius University, Vilnius, Lithuania, [3]also at CERN, Geneva, Switzerland

Abstract

The CMS Detector Control System (DCS) is implemented as a large distributed and redundant system, with applications interacting and sharing data in multiple ways. The CMS XML-RPC is a software toolkit implementing the standard Remote Procedure Call (RPC) protocol, using the Extensible Mark-up Language (XML) and a custom lightweight variant using the JavaScript Object Notation (JSON) to model, encode and expose resources through the Hypertext Transfer Protocol (HTTP). The CMS XML-RPC toolkit complies with the standard specification of the XML-RPC protocol that allows system developers to build collaborative software architectures with self-contained and reusable logic, and with encapsulation of well-defined processes. The implementation of this protocol introduces not only a powerful communication method to operate and exchange data with web-based applications, but also a new programming paradigm to design service-oriented software architectures within the CMS DCS domain. This paper presents details of the CMS XML-RPC implementation in WinCC Open Architecture (OA) Control Language using an object-oriented approach.

INTRODUCTION

CMS DCS applications are implemented using the SIMATIC WinCC OA [1] platform, mostly written in its native programming language called control (CTRL) language. The CTRL language is a C-like language with a very poor type definition syntax, not suitable for programming complex software architectures. The CMSfwClass [2] is a programming framework to build object oriented (OO) applications using CTRL language. The CMS XML-RPC toolkit relies on the CMSfwClass framework, which permits a better implementation of the software architecture after an extensive planning and design phase. The toolkit provides a set of software classes to build client-server architectures (See Fig. 1), enabling heterogeneous software entities to act as service providers (servers) or service requesters (clients).

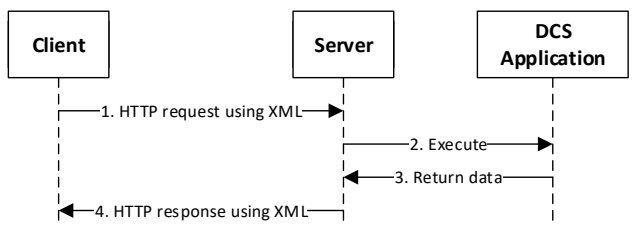


Figure 1: Client-server activity diagram on XML-RPC.

USE CASES

During the design phase, the software models were prepared to support at least the following scenarios: Remote procedure calls from the DCS user interface (UI), service-oriented collaboration between DCS applications, and DCS web services.

Remote Procedure Calls from the UIs

The CMS DCS is a large distributed environment, organized in a hierarchy of nodes and accessed from remote lo-

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP17

cations. In this context, the code associated to UIs is executed in the client machines while the events and data processing happens in the control nodes. Certain operations need access to resources that are only available in a remote control node (server). The CMS XML-RPC toolkit allows programmers to encapsulate server-only operations in the form of service objects that can be invoked from the UI. This feature allows the UIs to access resources, which are typically not accessible on the client side (E.g. local server files, system calls, etc.). In addition, the OO implementation hides the server details from the UIs; delimiting the concerns of the software and resulting in a cleaner implementation, compared to the classic procedural approach.

Service-oriented Collaboration

Many of the DCS applications are implemented as standalone applications, which are not ready to share functionality without further development. The CMS XML-RPC toolkit introduces a mechanism to model DCS functionality as a service. Different applications can be prepared to act as producers (servers) and consumers (clients) of services, permitting a well-structured collaboration interface between remote peers. In addition, this mechanism provides an abstraction layer to connect applications on different platforms.

DCS Web Services

The CMS XML-RPC toolkit complies with the standard XML-RPC specification using HTTP over TCP. This means that any application implementing these protocols can make use of the features exposed by DCS web services, including standard web applications. The CMS online portal (https://cmsonline.cern.ch) is a web portal hosting DCS related applications. Some of its applications connect to DCS web services; offering a new set of control features to the CMS users worldwide.

ARCHITECTURE

The CMS XML-RPC toolkit does not provide ad-hoc functionality. Instead, it provides a software model that can be used or extended to build new tailored client-server applications. The model includes two abstract classes describing the basic functionality of a service dispatcher and a service client. These classes are unaware of the final implementation details, as well as the message-encoding format, permitting the further extension of the architecture. In addition, the model includes concrete classes to build different client-server architectures.

Service Classes

The service classes are the core structures for building service based applications. The toolkit includes three subclasses derived from the abstract service class. Each of them extends the initial service implementation.

- Service class: This class implements the basic service functionality. Objects of this class offer a list of procedures to be executed remotely in the DCS context.
- Service router class: This class groups multiple service objects; offering a single entry point to clients. Objects of this class are particularly useful to concentrate and minimize the number of connections between the DCS and other platforms.
- Proxy server class: This class implement a message forwarding mechanism. Objects of this class can intercept, filter and divert requests to other service providers.

Service System Interface

The service system interface describes a list of popular RPC functions. The system interface is not part of the XML-RPC protocol itself, but it is a common feature supported by most of the available RPC servers. Using the system functions, clients can request the full list of available methods, their signatures or even the encapsulation of multiple calls to a remote server into a single request.

Client Classes

The CMS XML-RPC toolkit includes a set of service client classes, derived from an abstract service client class. Each of them implement different ways of invoking services.

- Client class: This class implements a basic method invocation by forwarding requests to specific service objects. Objects of this class use the internal WinCC OA run-time database to pass the messages to the service objects.
- Web client class: This class forwards the requests to a web service. Objects of this class listen to specific host ports using the transmission control protocol (TCP).
- Http client class: This class extends the web-service client functionality by using the HTTP protocol over TCP.

Proxy Server Configuration

By combining the server and client functionality into a single class, objects can configure a proxy server capable of serving, requesting, and therefore forwarding requests to other peers (see Fig. 2). A proxy server object can be configured to administer a group of resources, acting as a firewall between the CMS DCS web server and external clients.

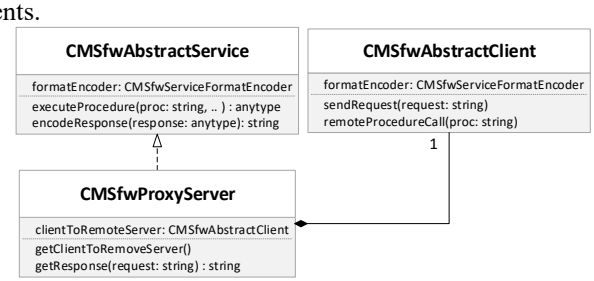


Figure 2: Service-client composition.

Message Formatting Classes

The information exchange between clients and servers require the encoding and decoding of messages. Clients should encode their request and transmit them using one of the available formats. Servers will decode the messages and process the request. The results are sent back to the

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP17

client, repeating the encoding-decoding operations on both sides. The CMS XML-RPC toolkit delegates the format encoding operations to objects implementing the message format. The different formatting classes implement the same interface; which defines the common methods for encoding and decoding messages. Thanks to this model, new formats can be easily included or extended without altering the final architecture.

IMPLEMENTATION DETAILS

The CMS XML-RPC toolkit is composed of 13 classes, 8 class interfaces and a Web service application handler. Once the framework is installed, objects can be created using a factory script or any of the available serialization mechanisms. The configuration of the objects and the relations between them is what determines the final software architecture. In total, the framework comprises around 3000 lines of code, in addition to the code required for building the final application.

Web Service Application Handler

WinCC OA provides an application programming interface (API) for building a standard HTTP server using CTRL language. The CMS XML-RPC toolkit uses the HTTP server API to implements a web-service application handler. Instances of any service class can be passed as parameters to the application handler to run a fully functional web service.

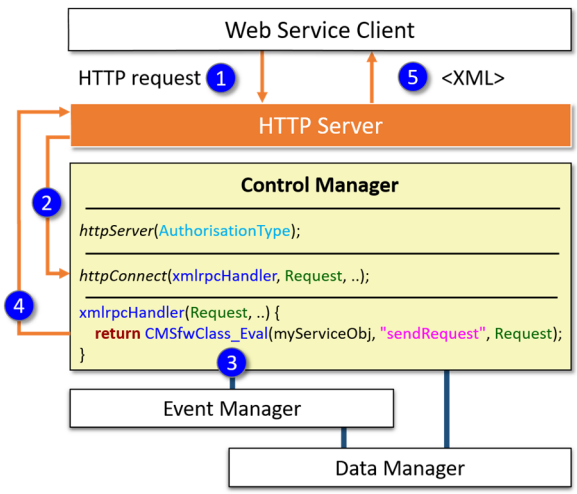


Figure 3: CMS XML-RPC application handler.

As shown in Fig. 3, web clients initiate the procedure with a HTTP request to the server. The application handler configures the HTTP server to listen to a particular port number in the control node. When the request arrives to the server, a call-back function processes it and delegates the execution to a service object. The service object decodes the request, executes the requested operation and encodes the results in the appropriate format. Results are passed back to the HTTP server, which replies to the corresponding client.

Message Format

The CMS XML-RPC toolkit provides two format-encoding classes. The first formatting class complies with the standard XML-RPC protocol specification. The second class implements a variant where responses are formulated in JSON. The usage of the JSON format is only available to web clients, which are able to decode this type of messages. This format is particularly useful in the context of JavaScript applications, since the encoding is part of the language notation. Messages encoded in JSON are lighter than the ones in XML (see table 1) requiring less space and processing time. Using one of the XML-RPC services, we have measured the length of the messages in the different formats (See Table 1).

Table 1: Size of the Messages

Type of message	Format	Bytes
Client request	XML	111 bytes
Server response	XML	1,042 bytes
Server response	JSON	597 bytes

A single query to the system returned a list of 25 items with 534 bytes of content data. The format of the XML message took more than 50% of the total message space. By contrast, the JSON format required only 1% of the message space. For this reason, JSON is the preferred format to exchange data with online applications.

INTEGRATION WITH CMS ONLINE

CMS Online is a web portal using the Oracle WebCenter Portal [3] technology to access technical information of the CMS experiment. The CMS online portal extends the DCS capabilities by offering a set of tools to monitor and administer parts of the DCS. Initially, the data exchange between the CMS Online and the DCS was limited by the usage of databases. Now, the CMS XML-RPC toolkit adds a new connection method to exchange data and perform remote operations from the CMS Online platform. At the moment, two web applications using the CMS XML-RPC toolkit are available:

- Online parametrization browser: This web application allows the users to inspect and change certain parameters in the control systems.
- Online DCS log files browser: This web application exposes the different logs available in the remote nodes, and implements some formatting and filtering features to facilitate the readability.

CONCLUSION

The implementation of the CMS XML-RPC toolkit started as a prototype to prove the feasibility of the XML-RPC protocol within the DCS domain. With the time, after several iterations to extend and refine its design, the toolkit became a consolidated part of the CMS DCS software. As result, the online applications based on the XML-RPC have also become an important tool for the CMS DCS community, allowing experts to access DCS technical information with the only help of a web browser.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP17

ACKNOWLEDGEMENTS

The authors would like to thank the Swiss National Science Foundation for the financial support.

We would like also to thank our colleague L. Masetti, who provided insight and expertise in the topics discussed in this paper.

REFERENCES

[1] SIMATIC WinCC OA [online], http://www.etm.at/index_e.asp

[2] R. Jimenez Estupiñan et al. "Enhancing the Detector Control System of the CMS Experiment with Object Oriented Modelling", ICALEPCS'15, Melbourne, Australia, October 2015, paper MOPGF025.

[3] Oracle WebCenter Portal [online], https://www.oracle.com/technetwork/middleware/webcenter/portal/overview/index.html

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP19

TINE RELEASE 5.0: A FIRST LOOK

P. Duval, J. Szczesny, T. Tempel, DESY, Hamburg, Germany
S. Weisse, DESY, Zeuthen, Germany
M. Nikolova, EMBL-Hamburg, Germany
J. Bobnar, Cosylab, Ljubljana, Slovenia

Abstract

The TINE [1] control system evolved in great part to meet the needs of controlling a large accelerator the size of HERA, where not only the size of the machine and efficient online data display and analysis were determining criteria, but also the seamless integration of many different platforms and programming languages. Although there has been continuous development and improvement during the operation of PETRA, it has now been 10 years since the last major release (version 4). Introducing a new major release necessarily implies a restructuring of the protocol headers and a tacit guarantee that it be compatible with its predecessors, as any logical deployment and upgrade strategy will entail operating in a mixed environment. We report here on the newest features of TINE Release 5.0 and on first experiences in its initial deployment.

INTRODUCTION

Originally a spin-off of the ISOLDE control system [2], TINE is both a mature control system, where a great deal of development has gone into the control system protocol itself, offering a multi-faceted and flexible API with many alternatives for solving data flow problems, and it is a modern control system, capable of being used with both cutting-edge and legacy technology. In addition to publish-subscribe and client-server transactions offered by many other control systems, TINE supports multi-casting and contract coercion [3]. As the TINE kernel is written in straight C and based on Berkeley sockets, it has been ported to most available operating systems. Java TINE, with all of its features, is written entirely in Java (i.e. no Java Native Interface). All other platforms, from .NET to Matlab to LabView to Python, make use of interoperability with the primary TINE kernel library. Furthermore, any client or server application based on TINE and its central services does not require any non-standard or third party software (i.e. there are no LDAP, MySQL, Oracle, Log4j, etc. dependencies).

The transition to TINE Release 4.0 was reported some time ago [4], where numerous features of TINE were enumerated, some of which (e.g. multicasting, redirection, structured data) set it apart from other control systems in common use. In addition, TINE offers a wide variety of features designed for efficient data transport and communication in large systems.

A series of meetings in 2012 identified long-term goals and established a roadmap for the future Release 5.0. Many of these goals have been realized over the past several years, showing up in new minor release versions of TINE, the last being version 4.6.3. What sets Release 5.0 apart and warrants a new major release number are some necessary changes to the protocol headers.

In the following we will identify and discuss those relevant embellishments which have ensued since the 2012 meetings and have culminated in TINE Release 5.0.

RELEASE 4 ISSUES

As noted in the introduction, a general collaboration meeting in 2012 identified certain aspects which needed to be addressed. These include the following.

Protocol Issues

The TINE protocol makes use of Berkeley sockets and TINE Release 4 originally did not properly support IP version 6 (IPv6), as the socket API calls used were all IPv4 centric. Although there is no mad rush to use IPv6, it does offer advantages which could be of interest in the not too distant future.

Header Issues

Several nice-to-have features, which potentially make life easier for administrators tracking connectivity problems, could only be added by expanding the existing protocol headers (and thereby requiring a new major release). For instance the process ID and application type of a connected client are not available under Release 4.

In addition, some supported features required workarounds under some circumstances, which could also only be ironed out by additional information not currently available in the Release 4 protocol headers. For instance, a generic client making a request to a server for a property's canonical data set can ask for the DEFAULT data set (and thus avoid an independent query to obtain the property characteristics). The returned data header will in fact provide the proper data format, but not explicitly give the correct data size. The latter can usually be inferred from the number of data bytes returned. However, if the request in question was truncated by the server, then the property data size which *should* be used in a request is an unknown quantity.

Finally, large data sets often require packet reassembly in the TINE kernel. For example, IPv4 jumbo datagrams can have a maximum length of 64 Kbytes. Any larger data set will require assembling multiple packets. In Release 4, the request and response headers hold the total message size in bytes in an unsigned short, i.e. precisely the 64 Kbytes of an IPv4 jumbo datagram. TINE transfers can of course use a TCP stream, or shared memory, rather than datagrams, but the same packet reassembly exists.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP19

That in itself is not a problem, except that it is often useful to specify a larger number for the message size in bytes, necessitating a 4-byte integer in the transport headers, rather than the current 2-byte integer.

Other Issues

A TINE server developer can choose among a variety of platforms on which to write his server, including Java, Python, LabView, Matlab, and .NET, not to mention the operating system. Nevertheless, a number of production servers are written in C or C++, making direct use of the C library API. C++ developers are most likely to make use of Standard Template Library (STL) or Microsoft Foundation Classes (MFC) libraries and headers. If this is indeed the case, then certain measures must be taken to avoid namespace collisions when *tine.h* is included in the same code module as the STL or MFC headers. This primarily has to do with macro definitions attempting to override e.g. a class name and cannot be trivially solved by using a namespace wrapper around *tine.h*.

RELEASE 5 SOLUTIONS

Protocol Issues

TINE Release 4.5.0 introduced the standard IPv6 socket API to the TINE library and by Release 4.6.3 the TINE libraries in both C and Java were fully implemented. The general strategy is for clients and servers to make use of a dual stack if possible, where a single bound listening socket can support either protocol. IPv4 clients will then only ever see an IPv4 address. Likewise an IPv6 client will always see an IPv6 address, be it a real one or a mapped IPv4 address (with a leading '::ffff:'. Thus this aspect was concluded prior to the advent of Release 5.0.

Aside from removing the administrative headaches involved in making use of private networks and exhausting the IPv4 address space, IPv6 also offers jumbo datagrams up to 4,294,967,295 bytes.

Header Issues

The TINE Release 5 request headers have indeed been modified to pass a client's process ID and application type to a server, along with associated diagnostics which pass this information along (see Figure 1). The application type is composed of an 8-character string identifying the principal kind of client making the call. A middle layer server acting as a client will supply the text "FEC", for instance, whereas a Python client will supply the text "PyTine". A client's process ID is perhaps of little or no use if the client is a command line tool such as *tget* used in a script. However, for persistent clients it is a useful identification number which can expedite the search for a specific client application should it become problematic.

The new response headers also categorically supply a contract's canonical data size as well as the size in bytes and in elements returned in the call.

Entities such as the message size or MTU are also now categorically 4-byte integers.

A contract response header also continues to supply additional system and user *stamps* as 4-byte unsigned integers. These are in addition to the associated data's time stamp, and are typically used to provide an event or cycle number tag to the associated data. That is, these were specifically *not* upgraded to 8-byte integers, primarily to avoid issues on 32-bit (or 16-bit) platforms which do not support them. A quantity such as an event number will wrap only every 14 years or so even if incremented at 10 Hz, so this should not present a problem in the short term, and will essentially never present a problem if the said *event number* is reset at the beginning of a run. In the long term, these quantities can be upgraded at some future time, should the need arise.

Both the request and response headers also provide information on the endianness of the host machine and the character encoding in use in the data provided. In the current release (5.0.0) the endianness is fixed as little endian and the character set is fixed as ASCII standard. One could argue that a modicum of efficiency could be squeezed from the system if only one of a client-server pair engaged in byte swapping when it needed to. However, as a Release 5 server will need to support Release 4 clients, which expect little endian payloads, it would have to make the decision to swap or not at the point of delivery. This is currently problematic for some data types, such as a user-defined tagged structure and some non-fixed length data types and requires extensive refactoring of code. Therefore this issue has been postponed for some future release, which of course will have to make the identical swapping decision based on a client having a release number greater than e.g. 5.2, etc.

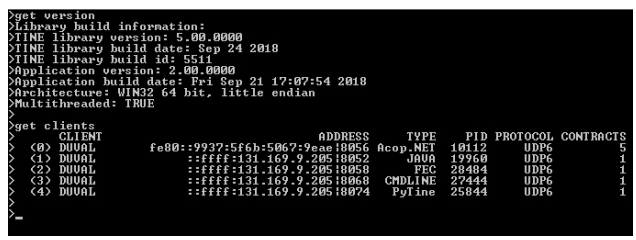


Figure 1: An example of a server's console command to show its attached clients. New to Release 5 are the PID and TYPE columns.

Other Issues

The problematic macro definitions (largely error/status codes) have essentially all be replaced with enumerations in Release 5. This has the advantage (as in the case of a macro definition) of not requiring constant variables in a program's data segment. Furthermore enumerations easily lend themselves to being used within a C++ namespace wrapper. Thus using the TINE API directly in C++ code no longer requires any extra measures to avoid namespace collisions. A namespace wrapper around the TINE header file *tine.h* is entirely optional.

UPGRADE STRATEGY

Any control system component making use of TINE Release 5 must be fully compatible with earlier releases

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP19

of TINE (predominately of Release 4 vintage). Release 5 servers must seamlessly interface with Release 4 (or Release 3) clients. And Release 5 clients must likewise be able to access earlier vintage servers. With this as an ansatz we can contemplate upgrading the control system elements adiabatically, with the expectation that legacy components will remain operational for months, if not years.

There is no *best moment* to roll out a new major release such as this, other than perhaps during a long shutdown, where there is often a prolonged re-animation of the machine. This happens infrequently. In any event, in the case of the PETRA III complex, no amount of unit testing will catch all compatibility issues, largely due to the multicultural aspects found in machine control there. For instance, there are critical Java servers running on both Windows and Linux hosts. There are 32-bit and 64-bit servers running not only on Windows and Linux hosts, but on VxWorks and LabView (also Windows) as well. Client applications are liable to be rich client Java applications using ACOP beans or jDDD (with its own complexities) or rich clients using Matlab, LabView, or using ACOP.NET [5].

As TINE is feature rich, there tends to be a wide variety of *ways to do things*. This in itself tends to increase the general entropy in a test environment.

Thus the path to general deployment was to test as much as possible, making use of the TINE unit server and client in combinations of Release 4 interfacing with Release 5, and then to deploy and react during the machine studies following a *mini* shutdown and prior to a user run. Here one can see which hiccups occur during normal operations and either rollback if necessary or find and fix (if the operators can tolerate the hiccup during a bit of extreme programing).

In fact, there were surprisingly few hiccups - three to be exact, two of which led to a rollback. Nonetheless, at the conclusion of the machine studies, TINE Release 5 was in place as the de-facto standard, although there will be a mixed scenario for some time to come. And to be sure, one still must continue to be on the lookout for hiccups and be ready to react.

LESSONS LEARNED

With the rollout of TINE Release 5, one generally hopes that, as far as the users and customers are concerned, *nobody notices anything*. That is to say, there are no new bells and whistles that would make transfers more efficient or offer new paradigms for application development. The API is basically unchanged. On the other hand, developers (especially server developers using the C library) will appreciate many of the new embellishments. Likewise administrators will find it easier to track communication problems.

The TINE core team will also be able to navigate through both the Java and C library code more easily, due to extensive refactoring. And as the latest TINE transport headers are extensible, it should prove to be a straightforward task to add fields to the existing headers at some time in the future should they be needed.

One somehow anticipates that by the mere act of shaking things up, i.e. not letting sleeping dogs lie, so to speak, various real problems (e.g. hidden race conditions) will be exposed. In the initial phase of the ensuing users run, two further upgrade issues in fact became apparent. One of these, a long-standing TCP issue which might occur when large input data sets are being collected at the server side, had *almost* no chance of expression in the Release-4 world and only became visible when the contract request headers increased in size in Release 5. This issue surfaced on a particular server and led to a local rollback until it was understood. The second issue involved a check on multi-channel contract coercion logic versus the minor release and revision numbers, which suddenly jumped back to 0 and 0. This latter issue had no *visible* consequences and was only noticed in that certain applications appeared to suffer in certain aspects of transfer efficiency. Both of these problems were promptly dealt with and neither had any direct bearing on the user run.

Introducing a new major release (or any systematic upgrade, for that matter) is not something one takes lightly at any time. In the absence of a full-blown mock facility which is actually used under *real* conditions, there is virtually no way to *catch things* other than to deploy and standby in extreme programming mode. All in all, there were surprisingly few hiccups due to specific software problems.

An additional hiccup was due to the non-synchronized deployment of system libraries. Although it had nothing to do with any software problems or Release 4/Release 5 compatibility issues, it would of course not have arisen had there been no attempt at an upgrade. Yet, this in itself exposed an existing problem (in this case, a misunderstanding in the software deployment on Windows hosts).

The moral of the story is: It sometimes takes a user run to expose problems! By now the dust has settled, so to speak, and one is gradually beginning to breathe more easily.

REFERENCES

[1] TINE website; http://tine.desy.de

[2] R. Billings *et al.*, "A PC Based Control System for the CERN ISOLDE Separators", in *Proc. ICALEPCS'91*, Tsukuba, Japan, Nov. 1991.

[3] P. Duval and S. Herb, "The TINE Control System Protocol: How to achieve high scalability and performance", in *Proc. PCaPAC'10,* Saskatoon, Canada, Oct. 2010, paper WECOAA02.

[4] P. Duval *et al.*, "TINE Release 4 in Operation", in *Proc. PCaPAC'08,* Ljubljana, Slovenia, Oct. 2008, paper MOX01.

[5] P. Duval *et al.*, "ACOP.NET: Not Just Another GUI Builder", presented at PCaPAC'18, Hsinchu, Taiwan, Oct. 2018, paper THCB1, this conference (and references therein).

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP21

INJECTION CONTROL OF THE TPS

J. Chen, Y. S. Cheng, C. Y. Wu, C. Y. Liao, K. H. Hu, K. T. Hsu
National Synchrotron Radiation Research Center, 30076 Hsinchu, Taiwan

Abstract

Injection control application served for Taiwan Photon Source (TPS) to help commissioning and operation of the machine. Top-up injection functionality is available from machine commissioning stage to accelerate vacuum conditioning. During last two years, several updates have been done to enhance flexibility for the injection control. The injection control includes foreground and background processes to coordinate the operation of e-gun, linear accelerator, booster synchrotron, storage ring by the help of event based timing system. Lifetime calculation of the storage ring is also synchronized with the injection process. Detail of the implementation will be presented in this report.

INTRODUCTION

The Taiwan Photon Source TPS [1] is a latest generation of high brightness synchrotron light source which is located at the National Synchrotron Radiation Research Center (NSRRC) in Taiwan. TPS consists of a 150 MeV electron linear accelerator (linac), a 3 GeV booster synchrotron, a 3 GeV storage ring, and experimental beamlines. Ground breaking for civil construction was held on February 2010. The civil construction completed in April 2013. Accelerator system's installation and integration started in later 2013. The control system environment readied in mid-2014 to support subsystem integration and test. After 4 months of hardware/software testing and improving, the commissioning of booster and storage ring was started in December 2014. First synchrotron light shines in the last day of 2014 [2].

Injection control application with top-up functionality was deploy at early period of commissioning phase to release loading of operator and increase beam dosage accumulation to accelerate vacuum conditioning. Constant current with lower and upper limit top-up injection was used for user service since September 2016. However, time interval between injections vary slightly which caused by beam lifetime variation affected by machine conditions such as coupling, nonlinear beam dynamics, insertion device parameters (gap and phase) changed, etc. User could hardly predict the timing of injection even hardware gating signal provided. The experimental data deteriorated by the injection transient. Some beamlines could use hardware gating signal to exclude the transient but most of the instruments and data acquisition systems which cannot gated by hardware gating signal were sensitive to the injection transient. The top-up injection control had been revised to fixed time interval between injections in early 2018. The time of next injection is predictable to suspend data acquisition during the injection window. Maximum stored beam current is set to 405 mA at the moment. The storage ring is refilled every 4 minutes. Beam current drops around 4 mA during this period, it takes about a few seconds (< 5 sec) to refill the stored beam current back to 405 mA.

INJECTION SUPPORT SYSTEM

The injection process need coordinate various subsystem such as electron gun, buncher, RF system of linear accelerator, injection and extraction pulse magnets of the booster synchrotron, and injection septa and injection kickers of the storage ring.

Linac and Booster Synchrotron

The TPS injector consists of a 150 MeV linac and a 3 GeV booster synchrotron. The 150 MeV linac was available for booster injection in the 2nd quarter of 2014. Measured linac beam parameters are well accepted with its specifications. The LTB transfer line has been successfully commissioned with beam and the magnet settings agree with designed expectation [3].

The booster synchrotron was commissioning in the third quarter of 2014 successful. Optimization were done in performance for routine operation. These optimizations consist of system upgrade to improve its weakness and the improvement of the ramping procedure to increase the capture and ramping efficiency of the beam charge, optics characterization [4].

Pulse Magnets and Pulsers

The pulse magnets of TPS consist of one booster injection septum and one booster injection kicker, two booster extraction septa and two booster extraction kickers, two storage ring injection septa and four storage ring injection kickers, and two storage ring pingers for diagnostic purpose [5]. Coordinate operation of these pulse magnets except pingers are necessary for storage ring injection.

Event Based Timing System

Event based timing system was adopted to coordinate operation of TPS include injection. The event system consists of event generator (EVG), event receivers (EVRs) and timing distribution fiber network [6]. Machine clocks (repetition rate, revolution clock of BR and SR, synch clock, etc.) are distributed by distributed bus. While trigger events for the injection are management by the sequence RAM inside of the EVG which installed the timing mater node which is the sources of major timing events. The 125 MHz event rate will deliver 8 nsec coarse timing resolution.

A control page for the EVG configure and testing as shown in Fig. 1, The interface provides a tool to configure event generator such as clock rate, sequence RAM, DBUS,

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP21

signal mapping, etc. To operate the accelerator efficiency, the injection control sequence was coded by state notation language (SNL) and execute by the EPICS sequencer running at the timing master as shown in Fig. 2. An injection interface page was implemented to provide a convenient interface to do beam injection for the storage ring. The injection sequence program running at the timing master EPICS IOC which communicate with the injection interface via process variables. Injection program is a state machine to help users to control the injection is shown in Fig. 3.

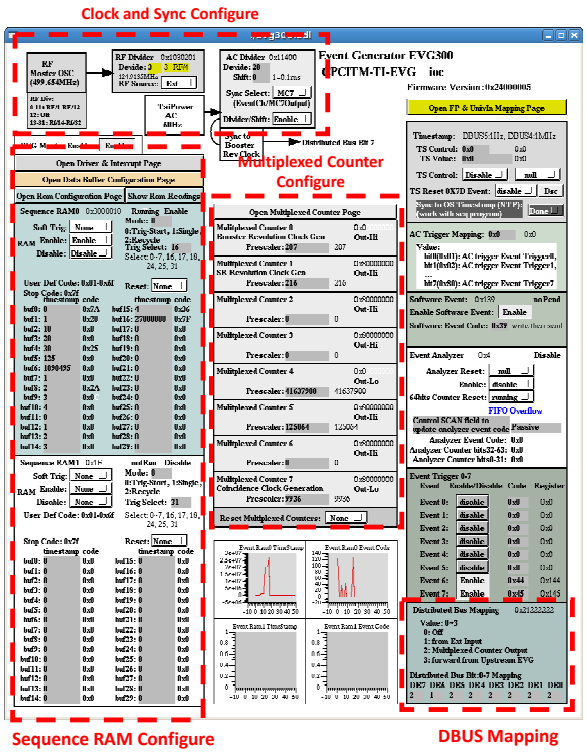


Figure 1: Event generator configuration graphical user interface.

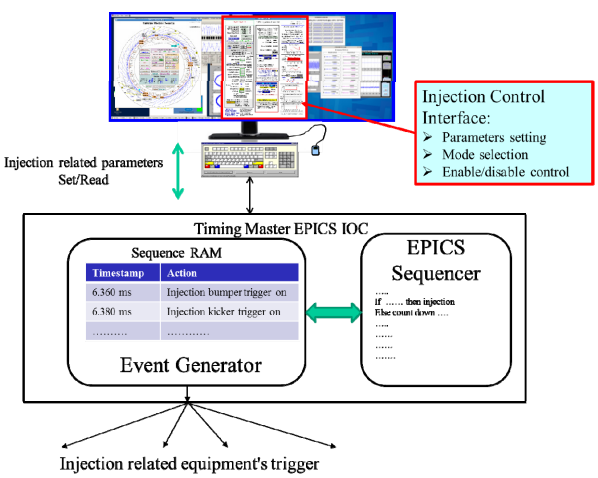


Figure 2: Injection control page communicate with the injection sequence program running in timing master EPICS IOC.

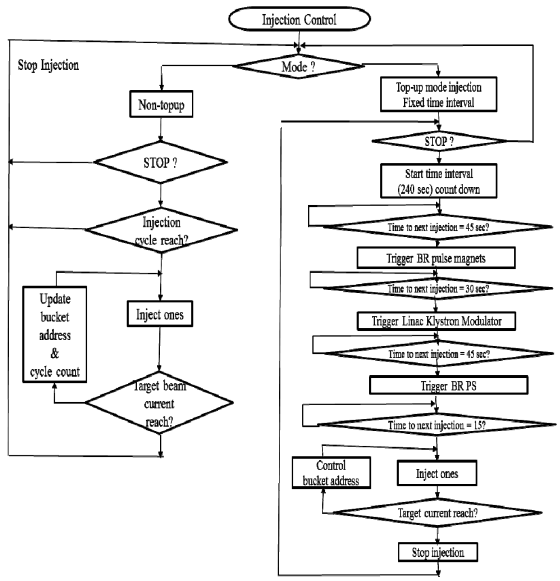


Figure 3: Sequence for injection control.

Filling Pattern Measurement

Filling pattern is an important factor which will affect storage ring operation performance such as lifetime, instability, etc. Keep the desired filling pattern are important during user operation which continue eleven-days operation every two weeks of current shift plans. Filling pattern should be monitor continuously. The acquired waveform from button signal by a dedicated oscilloscope with 40 GS/sec effective sampling rate trigger by revolution clock are used to extract bunch current with high resolution. The filling pattern analysis code extract bunch relative intensity accompany with total beam current measured by DCCT to calculate bunch current of individual bunch.

INJECTION INTERFACE

An injection control EDM page was implemented to support parameters configuration and operation of the injection as shown in Fig. 4. All of injection modes and sequences are managed by the content of sequence RAM which updated by the injection control EPICS sequencer program every injection cycle (0.333 second).

The left side of the page provide electron gun mode control, trigger enable, and dump beam setting. Timestamp stored in the sequence RAM is also listed.

Middle part of the injection control page consists of injection parameter setting, such as bucker address, injection cycle, target current setting, top-up control current range and time interval. The Top-up mode can select either multi-bunch mode or hybrid mode. If hybrid mode selected. it need specify the bucket address of isolated bunch. The injection process will be multiplexing the injection for multi-bunch and isolated bunch in hybrid mode as shown in Fig. 5.

Right parts provide synoptic display of beam current, filling pattern, beam lifetime, front-end status, and booster beam information.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP21

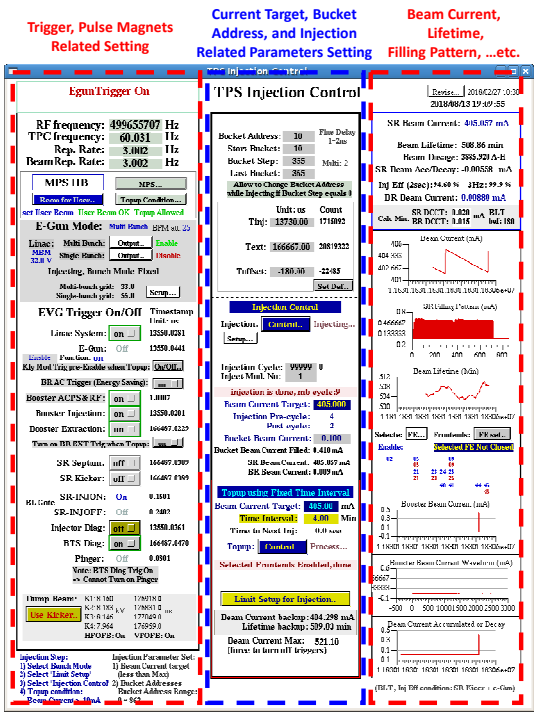


Figure 4: Injection control page.

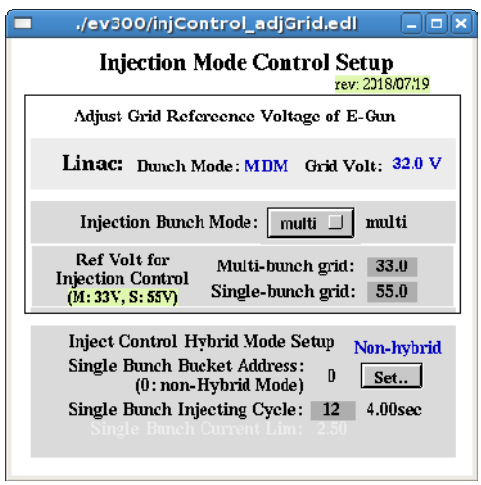


Figure 5: Top-up mode selection page.

Typical beam current and lifetime history, and filling pattern are shown in Fig. 6. Since the booster synchrotron can provide bunch-train near 1 μsec, current injection the electron gun and booster bunch train length as 700 nsec (350 bunches) in multi-bunch injection. The storage ring injection is only switch back and forth between two specific bucket location for bunch train injection until desired current accumulated. The gap of the lifetime data is due to lifetime calculation is stop during beam current accumulation. It need time to acquire sufficient data for regression before the first lifetime data available. So, the lifetime during this period is no updated.

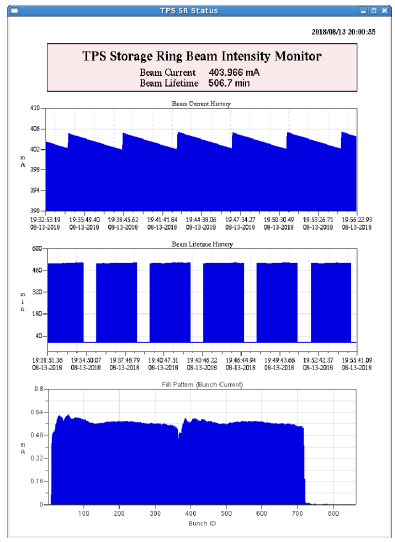


Figure 6: Beam current, lifetime, and filling pattern in typical user service shifts.

SUMMARY

Injection control for the TPS was implemented and revised several times since machine commissioning in late 2014. Event based timing system support synchronized action of various injection/extraction elements. Sequence RAM in event generator is used to configure the sequence of injection/extraction processes. Constant current mode was adopted from beginning of commissioning in late 2014 and user service started in September 2016. However, to help synchronize of experiment data acquisition with the injection process, constant time interval injection was chosen in early of 2018. The injection control functionality for TPS is working well and continue functionality revision will be performed to satisfy various requirements in future.

REFERENCES

[1] C. C. Kuo *et al.*, "Accelerator Physics Issues for TPS", in *Proc. IPAC'10*, Kyoto, Japan, May 2010, paper MOOCMH01, pp. 36-38.

[2] C. C. Kuo *et al.*, "Commissioning of the Taiwan Photon Source", in *Proc. IPAC'15*, Richmond, USA, May 2015, paper TUXC3, p. 1314.

[3] H. P. Chang *et al.*, "TPS Linac Relocation and Beam Test of the LTB Transfer Line", in *Proc. IPAC'15*, Richmond, USA, May 2015, paper TUPJE049, p.1731.

[4] H. J. Tsai *et al.*, "First Year Performance of the TPS Booster Ring", in *Proc. IPAC'16*, Busan, Korea, May 2016, paper MOPOR017, p. 634.

[5] C. Y. Wu *et al.*, "Control Interface of Pulse Magnet Power Supply for TPS Project", in *Proc. IPAC'15*, Richmond, USA, May 2015, paper MOPTY077, p.1120.

[6] C. Y. Wu *et al.*, "Integration of the Timing System for TPS", in *Proc. IPAC'14*, Dresden, Germany, June 2014, paper TUPRI113. p. 1833.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP22

THE LENS EFFECT IN THE SECONDARY EMISSION BASED SYSTEMS OF JOINT SEARCHING IN EBW*

A. Medvedev†, K. Blokhina, M. Fedotov, A. Starostenko, Y. Semenov, M. Sizov, A. Tsyganov
BINP SB RAS, 630090 Novosibirsk, Russia
A. Medvedev, BINP SB RAS, 630090 Novosibirsk, Russia

Abstract

The results of the developed scan lines generator for the magnetic correctors system are presented. Get the dependency between various types of scan lines and distribution of the allocated energy in the electron beam welding facility. The lens effect in the secondary emission based system of joint searching, using 3-fragment linear scan line is received. The accuracy of the joint searching system (the error of the positioning system) is 0.05 mm, the lens effect can decrease this value several times. The requirements for the creation full calibrated system of joint searching are listed.

INTRODUCTION

Electron beam technologies based on using beam source (electron gun) with accelerating voltage about some decades kilovolts and power from several watts to several decades kilowatts. The joint search system is very important part of the electron beam welding facility, it allows scan the surface of the target and sees the map of scanning area with asperities of the surface, in particular, the system allows see the joint.

Basically, the principle of functioning the joint search system is reflection and reemission the part of electrons, which interact with the target surface. The back current is measured by the isolated electrode. Different areas of the surface have different reflection coefficient, so we can see the different current values depending on the surface traits. In practice, we need the beam reflecting system with a reading of values from sensors.

EXPERIMENTAL SETUP

The experiments were conducted in the Budker Institute of Nuclear Physics (Siberian Branch of the Russian Academy of Science). The experimental setup has the vacuum chamber, the electron gun (up to 60 kV) with current ability up to 750 mA [1], two-coordinate reflection system based on cos(theta) coils, magnetic amplifiers (bipolar current sources up to 1.5 amperes), electrode, amplifier, and processing block of the joint search system.

Processing block include several channels of digital-analog converter which form the signal for magnetic system, one channel of analog-digital converter, which measure the value from the sensor, FPGA based device for synchronous communication between DAC and ADC, and the single-board computer, which provides the network access for staff of the facility, and realizes the interface to change settings of scanning (by changing internal values in FPGA). Figure 1 shows the main equipment of joint searching system, and interaction point with the rest of facility equipment.

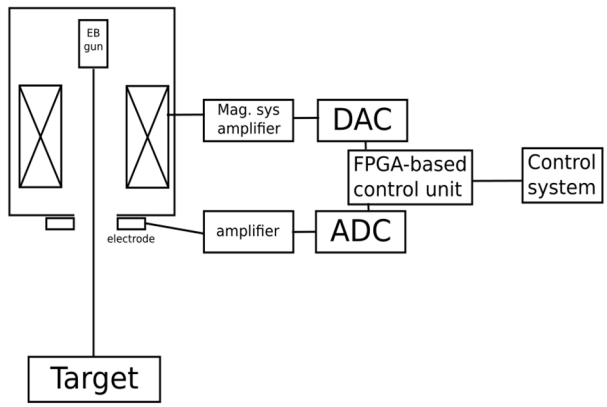


Figure 1: Scheme of the joint searing system.

THE SCANNING PROCEDURE

Despite the scanning system has two coordinates, usually for practice purpose we need only one of them. The beam intersects the joint each period of scanning, and the target movement system moves the detail along the joint. As a result, we can see the position error for all length of the joint. The dual scan uses when we need to get the frame of the scanning area. The experimental setup allows creating this type of scan.

We can illustrate the signal formed by the scanning system. In the simplest case, this signal is sawtooth. Usually, we use a sawtooth-like signal with positive and negative ramps (triangle wave). Using of the one-side sawtooth signal may give us negative effects with quality and appearance of the seam (in the case, when we weld the details with scan), so we use symmetric types of scan.

The scanning system forms the map of the surface, and the size of the map corresponds to the size of the scanning area. We can change the amplitude of the scan, depending on the view, which we want to see. It is possible, to get the ratio between scan amplitude and width of the scanning area. Due to the incline of the beam, the ratio will be changing for each new vertical position of the target. This problem can be solved by using 2 magnetic correctors, one above the other, with the birefringence of the beam. This moment we have the second pair of magnetic correctors but haven't made hardware and software modifications in the control system to get this opportunity. So we fix the distance between the beam gun and target in order to get

* supported by Russian Science Foundation (project N14-50-00080)
† alexey.m.medvedev@gmail.com

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP22

repeatability of experiments (approximately 100 mm from the end of the gun).

The quality of the plotted map depends on the noises level in reflected beam current, the geometry of the collector electrode, and the amplification factor. Besides, we can use programming methods to increase the quality of the image. One of them is filtering methods applying to the big set of data measured by special function in automatic mode. The user interface allows seeing the map in real time, with noises. It is very useful for a primary setting of the target position.

THE LENS EFFECT

The architecture of the joint searching system allows setting any data sequence to form the signal for the magnetic system. We develop the 3-stage linear signal, with the ability to set different incline and different size of each stage. Two parameters set the form of signal, and the program generates it automatically. The most interesting form shown on the Fig. 2. The beam moved with small velocity on the central part of the scan line and has high velocity at the edges of the scan line. The same time, the surface map shows the signal evenly in time, so the central part of the scan line has a bigger segment in the surface map. We named this "the lens effect".

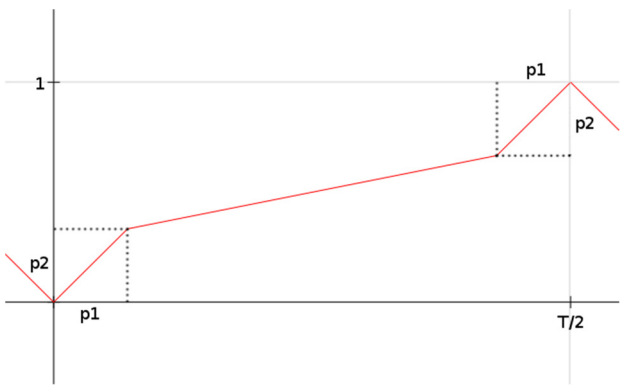


Figure 2: Three-stage linear signal.

The parameters of this digital lens could be adjusted by the scan line generator. The shape could be transformed in various waves with different lens effect (from triangle wave to straight line or square wave).

The shape of the signal influences to power distribution in the welding application. The form with lens effect in the central part of the scanning area increases energy release in the joint. Thus the depth of the seam is increasing and we can improve welding quality and weld with less beam current.

ACHIEVABLE ACCURACY

This section describes the boundaries and restrictions of the joint search system and says about accuracy, which we can reach on the facility. First, the beam has it's own size, the lens of magnetic optic system of the beam could reduce the size of the cross-section of the beam, but the minimum size is limited by gun geometry, beam quality and influence of space charge of the beam. Gun of BINP EBW facility has beam minimum size about 0.5 mm (with current used for scanning).

Second, the control signal is formed by 14 bit DAC. The amplitude of the beam trace with using maximum range of the DAC is 2 cm (could be increased by using another model of DAC). One LSB corresponds to 1.25 um of beam trace; this is the extreme achievable value of minimal beam movement. For scan shape, we use an array of forming a line with 500 values. The length of the trace segment between two neighboring values with maximum scan amplitude is 40 um. By reducing the amplitude or by using the mode with lens effect we can increase the resolution.

Besides the spatial properties, there are temporal characteristics of the scanning system. The frequency of the scan is limited by Foucault currents and capabilities of the DAC. Practically, we can achieve 1 kHz frequency of the scan.

CONCLUSION

We developed the joint search system based on secondary emission effect. Possibility to set any form of scan line allows getting the scanning system with lens effect. The minimum size the system can show, limited by the beam size, on the electron beam welding facility this value is about a one-tenth millimeter.

ACKNOWLEDGEMENTS

This work has been supported by Russian Science Foundation (project N14-50-00080).

REFERENCES

[1] Yu. I. Semenov, P. V. Logatchev *et al.*, “60 keV 30 kW electron beam facility for electron beam technology”, in *Proc. EPAC’08*, Genoa, Italy, Dec. 2008, paper TUPP161.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP23

CONTROL SYSTEM USING EPICS TOOLS AT TARLA LINAC

O.F. Elcim†, Institute of Accelerator Technologies, Ankara U., Ankara, Turkey

Abstract

The first accelerator based research facility of Turkey-TARLA is under commissioning at Institute of Accelerator Technologies of Ankara University. It is designed to generate free electron laser and Bremsstrahlung radiation using up to 40 MeV continuous wave (CW) electron beam. The control system of TARLA is based on EPICS and are being tested offline. TARLA also has industrial control systems such as PLC based cryoplant and water cooling system. Its control system is under development, it benefits from the latest version of EPICS framework, i.e. V7. In other words, TARLA control system uses existing demonstrated tools of EPICSV3 as well as pvAccess which comes with EPICSv4 for transferring the large data through control network. Archive (CSS BEAUTY) and alarm (CSS BEAST) system have been set up to detect stability and prevent failures. Operator interfaces have been designed using CSS BOY. Currently, CCDs, PSS (Personel Safety System), MPS (Machine Protection System), Superconductive Cavities, RF Amplifiers, microTCA based LLRF system are being integrated into distributed control system. In this proceeding we summarize the current status and future plans of TARLA control system.

TARLA CONTROL SYSTEM

Control system is one of the main issues. TARLA approach so far was to have sub-systems as independent as possible (including all business logic) with local control system while exposing status read-backs and configuration parameters via TCP/IP interface (slow control only).

TARLA LINAC control will be performed by an EPICS based system operating under the Centos7 operating system on industrial rack mount. EPICS was selected as the main medium due to its commitment to new era accelerators as well as distributed structure of architecture and high performance characteristics. The latest addition to EPICS base, EPICS V4 [1] (EPICS 7 was released in August 2017), was also included in the design goals.

The development environment already used:

- CentOS 7
- EPICS base 3.15
- EPICS V4 4.6
- CSS 4.5.6 (basic)
- MySQL
- Git

Hardware platform for slow control is not finalized. United Electronics hardware platform is being used as IOC for analogue and digital interfaces:

- DNR-6-1G: Compact (3U), 6-slot, rugged, Gigabit Ethernet Data Acquisition and control rack
- UEIPAC 600R: Real-Time, GigE, programmable automation controller (PAC) with 6 I/O slots
- UEPAC 300-1G: Real-Time, GigE, programmable automation controller (PAC) with 6 I/O slots

"In-house" developed TCP/IP based general purposed I/O cards or other custom developed hardware will also be used, while product from other manufactures are also possible.

Standard IOC for devices with TCP/IP interfaces is MOXA DA-682A, x886 2U rackmount computer with 6 gigabit Ethernet ports and 2 PCI expansion slots.

When designing the TARLA Control System, priority and purpose is to create an easy to use/maintain, soft IOC-style, fast, stable and extendable control system suitable for the control and monitoring requirements of all auxiliary systems as well as for laser creation and beam diagnostics.

Architectural Design

Control systems for experimental physics facilities are usually structured in three tiers (see Fig. 1):

User Interfaces. These can be either graphical or non-graphical (command line based) and are usually located in the control room. However, there could be some user interfaces used elsewhere during commissioning and maintenance.

Central Systems and Services. Systems like Timing, Machine and Person safety and computer services that need to run continuously irrespective of user activities, e.g., for archiving acquired data, monitoring alarm states, user authentication services etc.

Equipment Interfaces. This tier is responsible for interaction with equipment and devices. It provides an abstract representation of equipment to higher layers through which the equipment can be monitored and controlled.

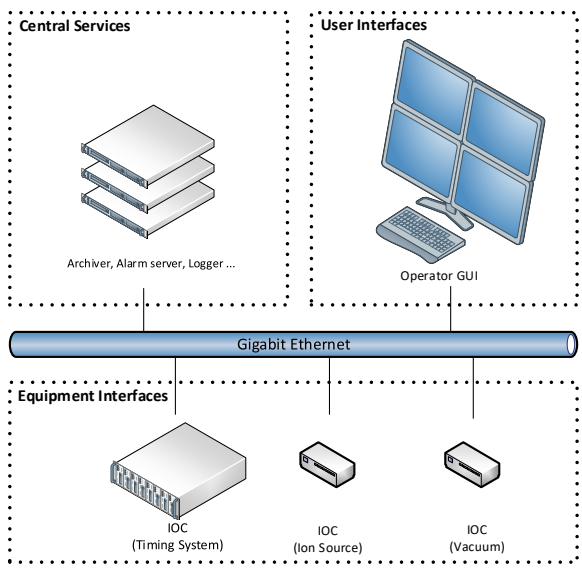


Figure 1: Three tier structure in TARLA control system.

∗ Work supported by Ministry of Development of Turkey
† omer.faruk.elcim@tarla.org.tr

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP23

A vertical column (see Fig. 2) is an abstraction which describes how a concrete device is integrated into the control system. It covers everything from the device sensors to the GUI on the operator machine.

It can be viewed from the two perspectives, presented on EPICS example in Figure 2:

- Looking at hardware it follows signal from the sensors/actuators through device controller to the computer running IOC (EPICS server) and finally to the operator's workstation (EPICS client).
- Software vertical column defines operating system(s), device drivers, required EPICS modules, device specific GUIs etc.

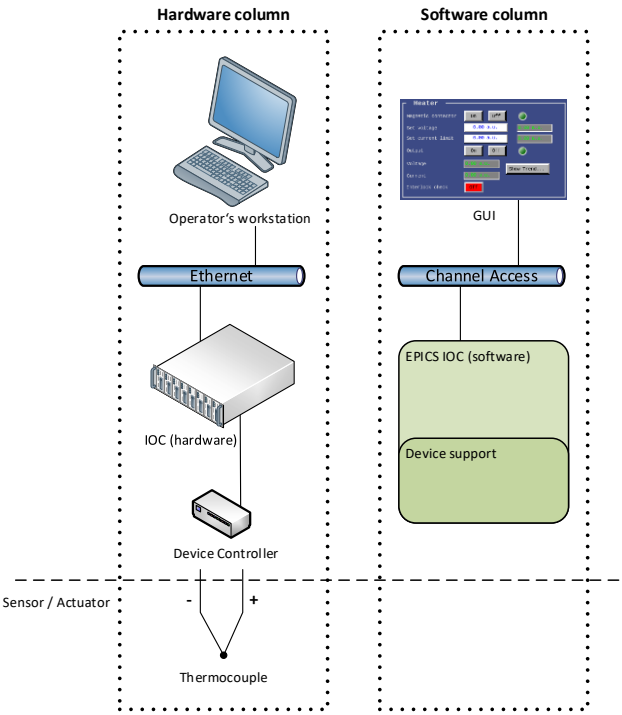


Figure 2: Hardware and software vertical column in EPICS.

Alarm Handler

Alarms are special kind of messages that indicate conditions that need operator awareness and additional actions, supervised by operator. Alarms may range in criticality level.

Alarms are hierarchal structured into levels. Number of levels should be limited and criticality of each level documented, considering the potential effect of the underlying cause of the alarm on operational runtime, equipment damage and safety of personnel.

An alarm server should be responsible for collecting all alarms, storing them in a dedicated database and dispatching them to the operator display.

Most promising alarm server and viewer in EPICS community is BEAST, which is important part of the Control System Studio [2], which is also used in TARLA Control System (see Fig.3).

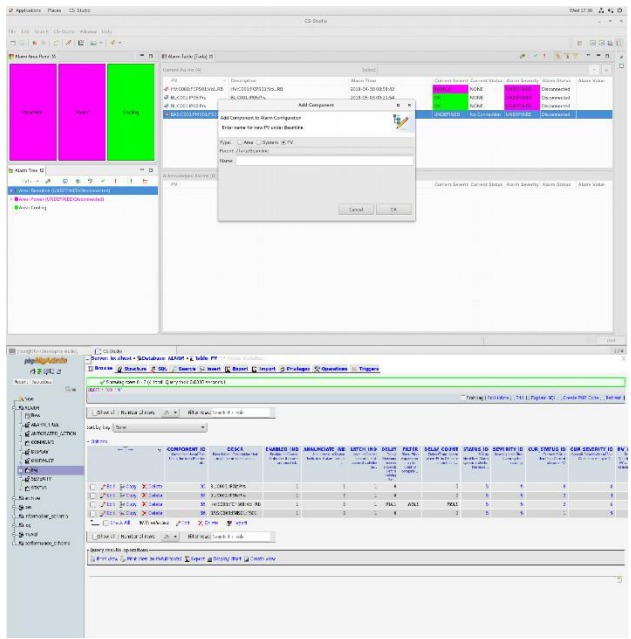


Figure 3: Alarm handler of TARLA control system.

Archive Engine

Providing historical data about the linac operations is up to archiving service. This data is used to evaluate TARLA linac performance, diagnose issues and failures and establish correlations between measurements and several settings for tests. It is important to design of archiving application and the accessibility of the archived data, they will heavily affect machine diagnostic efficiently.

Archiver service for TARLA is EPICS Archiver Appliance (see Fig. 4), which is among others used by ESS and SLAC. It stores data in the binary files on a hard drive, SSD or RAM.

The archiver appliance is written in Java and can run on most common Linux distributions.

One of the main advantages of the Archiver appliance is the built-in support for data storage on short term storage, medium term storage and long term storage.

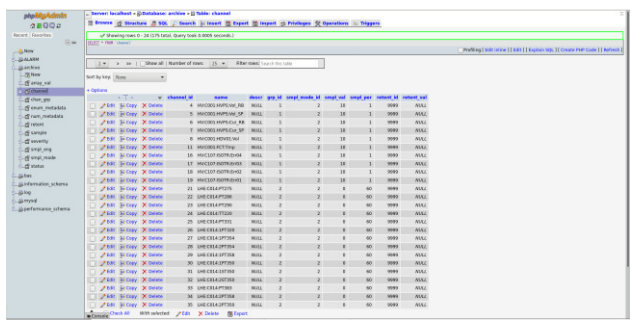


Figure 4: Archive engine of TARLA control system.

Viewer for the archived data is "Data Browser" perspective in the Control System Studio (see Fig. 5). Data can be retrieved from the Archiver Appliance to the Control System Studio with out of the box support.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP23

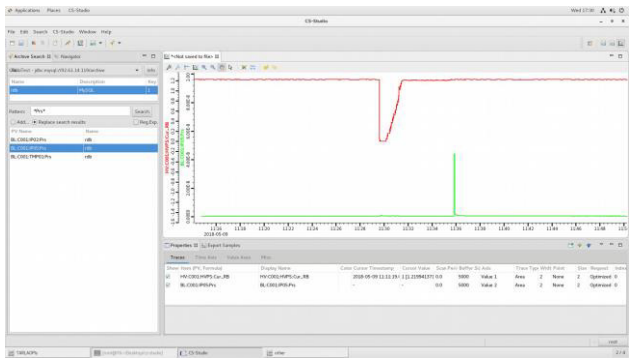


Figure 5: CSS data browser, viewer of archived data.

IocLogServer

Logging provides engineers and developers with insight into the current behaviour of TARLA CS components. It is an important tool for diagnostics when TARLA control system's functionality does not fit with set necessities and allow developers and engineers to resolve problems.

The logging service should collect time-stamped log entries from the control system and store them in a central location, where efficient queries can be performed.

EPICS already provide services for logging such as "iocLogServer" and "iocLogClient" components. Apart from EPICS logs other log files generated by computer nodes (e.g. Linux log files) can also be centrally managed with off the shelf services such a "syslog". Logs provided by the "iocLogServer" are text based.

JAVA-based software for saving IOC's error records to the RDB directory. The modified version of EPICS iocLogServer can read in configuration file in start-up time. This configuration file specifies how the server must process messages from particular IOC (files where messages must be saved, log file limits etc.). Flexible format of configuration file allows to define different methods of message processing. Now three methods are implemented: basic, non-stop and monthly. Each IOC can have its own configuration[3].

OLOG Logbook

Olog is implemented as a REST style web service. Its intended use is as an electronic logbook for accelerator operations [4].

It is a JAVA-based application that enables practical observation of important observations, notes, break-downs.

Control System Studio has an interface for viewing OLOG logs, which can be accessible from path Window > Open Perspective > Other < Log Viewer Perspective (see Fig 6.).

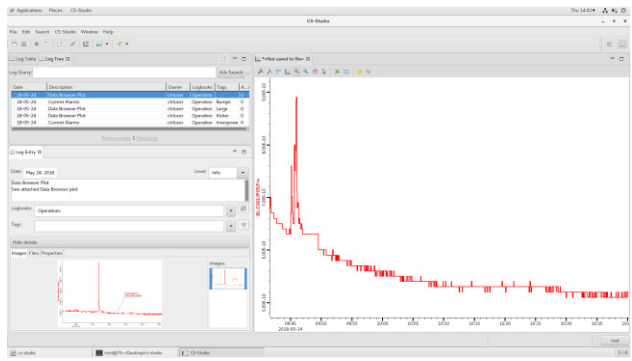


Figure 6: OLOG CSS interface.

CONCLUSION

Control system for TARLA e-gun is capable of producing and controlling the electron beam in DC/CW or macro-pulsed mode at the moment[5].

Device integrations to the EPICS-based control system and infrastructure improvements of the control system continue parallel to the installation of the line. The integration of the injector line control and diagnostic devices of the linac will be completed in the first quarter of 2019.

The purpose of the TARLA team as well as control system group is to build a state-of-art system, fully featured and debugged by the first laser beam generation.

REFERENCES

[1] http://epics-pvdata.sourceforge.net/index.html

[2] http://cs-studio.sourceforge.net/docbook/

[3] http://www-mks2.desy.de/

[4] http://olog.github.io

[5] Kazanci *et al.*, "Current Status of TARLA Control System", in *Proc. IPAC'14*, Dresden, Germany, Jun. 2014, paper THPRO127.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP24

THE STATE MACHINE BASED AUTOMATIC CONDITIONING APPLICATION FOR PITZ

D. Melkumyan*, P. Boonpornprasert, Y. Chen, J. Good, M. Gross, H. Huck, I. Isaev, D. Kalantaryan, M. Krasilnikov, O. Lishilin, G. Loisch, A. Oppelt, M. Otevrel[1], B. Petrosyan, H. Qian, Y. Renier[2], F. Stephan, G. Trowitzsch, G. Vashchenko, DESY, 15738 Zeuthen, Germany

[1]also at CEITEC, Brno University of Technology, 612 00 Brno, Czech Republic
[2]also at DPNC, Geneva University, 1211 Geneva, Switzerland

Abstract

The Photo Injector Test Facility at DESY in Zeuthen (PITZ) was built to test and to optimize high brightness electron sources for Free-Electron Lasers (FELs). In order to achieve high accelerating gradients and long RF pulse lengths in the normal conducting RF gun cavities, an extensive and safe RF conditioning is required. A State Machine based Automatic Conditioning application (SMAC) was developed to automate the RF conditioning processes, allowing for greater efficiency and performance optimization. The SMAC application has been successfully applied to RF conditioning of several gun cavities at PITZ.

INTRODUCTION

The PITZ [1] has been established to develop, test and optimize sources of high brightness electron beams for FELs like FLASH [2] and the European XFEL [3]. Essential requirements of an electron injector for FELs are the ability to generate a reliable electron beam with a very small transverse emittance (e.g. <1mm-mrad, 1nC bunch charge) and a reasonably small longitudinal emittance. The primary goal at PITZ was to facilitate stable and reliable operation with 60 MV/m accelerating gradient at the photo cathode at 650µs RF pulse length and 10Hz repetition rate. To achieve this, a new RF feed system with two RF windows was installed at PITZ in 2014.

The RF Setup

The gun prototype 4.5 conditioning setup consisting of a 10-MW multi-beam klystron, an upgraded RF waveguide distribution system with SF_6 and Air parts, two 10-MW THALES [4] vacuum RF windows, directional couplers, Ion Getter vacuum Pumps (IGP) and a Pressure Gage (PG), photomultipliers (PMT) and electron detectors (e-det) located around the gun coupler is shown in Figure 1.

The Gun

The PITZ photo electron gun is a 1.6 cell normal conducting L-band cavity, with cathode located at the back wall of the half-cell. The electron beam is generated at the cathode by a laser pulse and then accelerated by RF fields and focused by the solenoid fields.

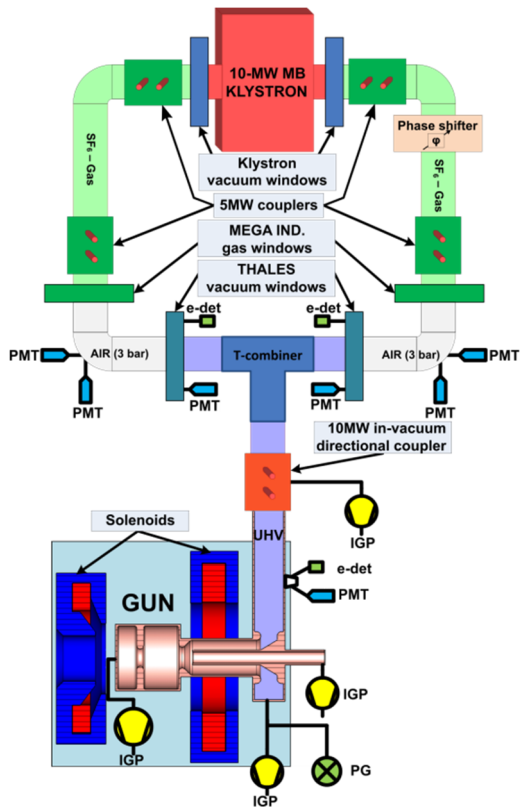


Figure 1: The PITZ RF setup (version 4.5, 2018).

Conditioning

The main goal of the conditioning process is to improve the vacuum in the cavity and the surface condition of the cavity, while applying the RF power is slowly increased in steps up to the operating gradient. During the conditioning process breakdowns may happen with a sharp rise of the pressure in the cavity. In such event the voltage in the cavity is dropped to some initial value and then slowly increased again. The final goal of the gun conditioning is to achieve stable operation at the maximum power level in the gun with solenoid fields.

The State Machine based Automatic Conditioning application (SMAC) was developed to automate the conditioning process for the RF cavities. The application was designed to replicate the operator behaviour during conditioning an RF cavity, allowing for greater efficiency and performance optimization. The interface for the SMAC application has been designed to be user-friendly and intuitive. It provides the user control of the conditioning process and relevant monitoring data.

* david.melkumyan@desy.de

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP24

THE STATE MACHINE BASED AUTOMATIC CONDITIONING APPLICATION

SMAC is written in Java and uses State Chart XML (SCXML) [5-7] as the finite-state machine execution environment based on Harel state-charts [6]. It employs the Distributed Object-Oriented Control System (DOOCS) [8] and Three-fold Integrated Networking Environment (TINE) [9] for the communication with the control systems of PITZ such as RF System, Interlock System [10], Alarm system, Water Cooling System (WCS), etc. The graphical user interface (GUI) is created by using the Java Swing toolkit and available via Java Web Start (JWS) [11] which provides a simple way to launch an application via a network. Communication between GUI and SXML processing layer is performed via Document Object Model [12] (DOM) events. A pure Java design makes the application extremely portable across computing environments. In the end SMAC produces very detailed logs in order to see how the workflow has progressed from step to step and observe any errors that may have occurred. The overall structure and data flow of SMAC application is shown in Figure 2.

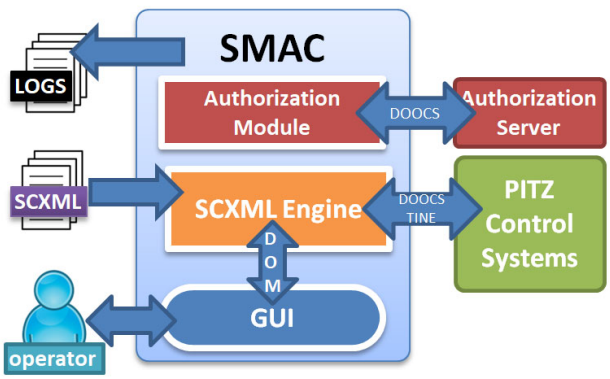


Figure 2: Overall structure and data flow of SMAC.

The Authorization Module

The authorization module with a help of the Authorization server guaranties that only one instance of SMAC is working at the same time. This will prevent concurrent conditioning processes that would lead to application malfunction and even cavity damage.

The SCXML Engine

SCXML engine capable of executing a state machine defined using a SCXML document that describes the application flow. Figure 3 shows a SCXML file segment that defines top-level states of the RF conditioning process. The main advantages of the SCXML approach are listed below:

- Legibility: the whole application flow is easily visible by looking at the flow definition files.
- Flexibility: using only the SCXML document the application control logic can be extracted out of the source code and specify very modular components into an SCXML.
- Standards-based solution: specified by W3C the SCXML is becoming a standard way to represent state charts in XML. It is part of Unified Modeling Language (UML) [13].
- Any SCXML can be easily parsed and interpreted by other software. An SCXML document can be created even using various graphical tools.

```
<state id="rf_configure" src="../conf/rf.config.sc.xml">
<state id="idle_rf_ramping">
<parallel id="process_rf_ramping">
<state id="error_rf_ramping">
<state id="interrupted_rf_ramping">
    <onentry>
        <log label="interrupted_rf_ramping" expr="'RF ramping is in
    </onentry>
    <transition event="resume.rf.ramping" target="process_rf_rampin
    <transition event="terminate.rf.ramping" target="terminated_rf_
</state>
<state id="terminated_rf_ramping">
```

Figure 3: Segment of SCXML code.

The GUI

Figure 4 shows a screen snapshot of the SMACs interface whilst running.

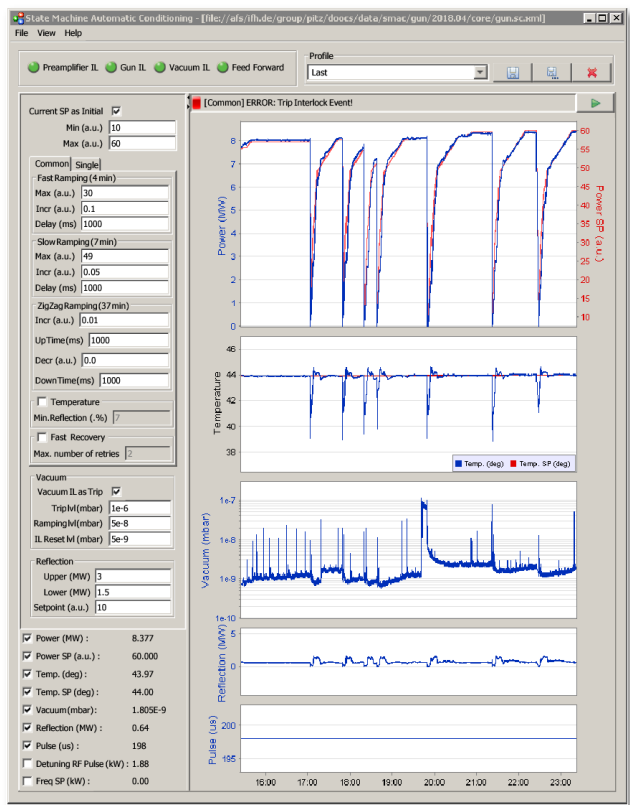


Figure 4: A screenshot of the GUI.

The panel on top-left shows the feed-forward signal and the Interlock system statuses. The profile panel on the top right allows operator to pre-configure the conditioning settings in order to quickly apply them to a new run. The plots on the right present the history of the RF power, water temperature, vacuum pressure, reflected power and the RF pulse length. The bottom left displays the current values of monitored parameters. Parameters of the condi-

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP24

tioning procedure can be set at the middle left of the screen. The main parameters set by the user are:

- RF power range and ramping speeds.
- Vacuum upper/lower thresholds.
- Reflected power thresholds.
- Enabling WCS temperature control.
- Enabling "Fast Recovery" [14] procedure after each breakdown.

Default values for these parameters were set based upon prior experience of the operators.

CONDITIONING ALGORITHM

The conditioning algorithm consists of gradually increasing the RF power and the RF pulse length but keeping a low rate of vacuum spikes in the cavity in order to prevent any damage from breakdowns. The algorithm based on conditioning requirement of the THALES windows and adopted for the gun cavity conditioning. It follows simple rules:

- RF pulse length: 10 – 650μs.
- Vacuum pressure: below 10^{-7} mbar.
- RF power ramping speed: maximum 0.2MW every 15 minutes for each new RF pulse length, when the RF window has seen this power level the first time.
- In the case of significant vacuum spikes or breakdowns:
  - Reset the RF power to initial value.
  - Reset pulse length to minimum value (10μs).
  - After reaching the maximum RF power, increase the pulse length in reasonable steps.
- Initially, the RF gun solenoid is off.
- Achieve the maximum RF power level with a negligible breakdown rate (aim for less than one breakdown per week).
- Finally repeat the conditioning procedure with sweeping RF gun solenoid currents.

However such a conditioning algorithm might be complicated since it involves simultaneous monitoring and controlling various operating parameters (e.g. interlock/feedforward/feedback statuses, RF power, gun temperature, RF frequency, etc.). Figure 5 shows a segment of the RF ramping flow diagram.

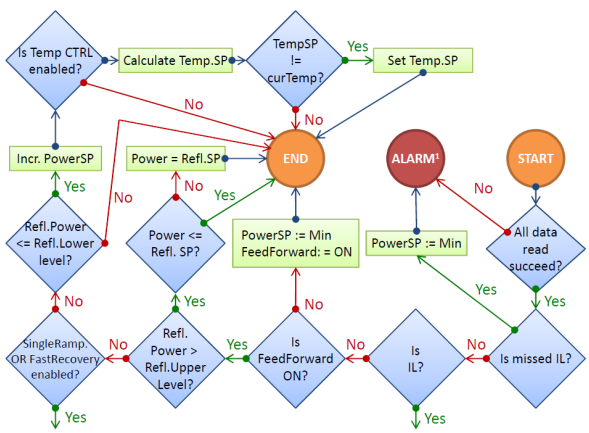


Figure 5: Segment of the RF ramping flow diagram.

The current version of the SMAC application implements two ramping modes:

1. Single mode: based on the "Fast Ramping" algorithm [14]. Thus the RF frequency is changed to follow the resonance frequency and the temperature of the gun cavity is continuously adjusted for slightly overheated operation. This process is continued until a certain power level is reached or a breakdown occurs.
2. Common (or continues) mode is shown in Figure 6: RF power is steadily increased until a significant vacuum spike or until breakdown occurs. The common mode consists of several stages ("fast", "slow" and "zigzag") with different RF ramping speeds. In case of relative high level of pressure in the cavity or high level of RF reflected power:
   - Suspends increasing of RF power until suitable ramping conditions.
   - Decreases RF power to initial value, if reflected power or pressure in the cavity is too high.
   - After each breakdown SMAC attempts to reach the last power level applying the Fast Recovery procedure if enabled.

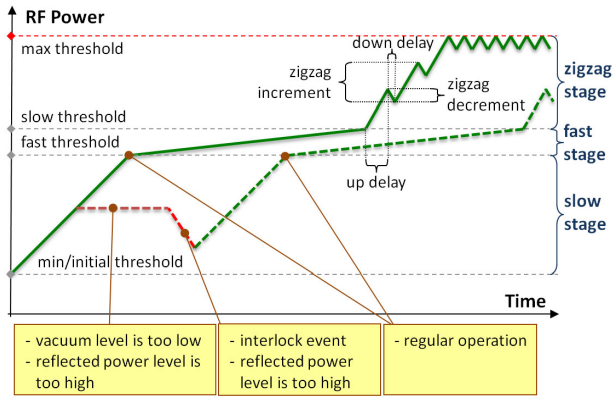


Figure 6: Common mode of the RF power ramping.

CONCLUSION

In this paper, we have presented the main design features and the current implementation status of the SMAC application. This implementation was intended as a proof of concept, applying a state-chart approach toward the development of the automatic conditioning application. The state-chart approach has been found particularly useful because of its appealing hierarchical, communication and concurrency constructs. The use of Java SCXML engine was very encouraging and we intent to explore the use of it with advantages offered by the Java technology. The SMAC application was brought into operation in 2010 and has been used at PITZ very successfully for all RF cavities. Since then the application is left to run unattended overnight. The RF conditioning strategy is based on recommendations from physicists of the PITZ group. The SMAC continues to be improved by feedbacks and suggestions from the physicists.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP24

REFERENCES

[1] PITZ, "Photo Injector Test Facility at DESY," http://pitz.desy.de

[2] W. Ackermann *et al*., "Operation of a free-electron laser from the extreme ultraviolet to the water window," Nature Photonics, vol. 1, pp. 336-342, 2007.

[3] M. Altarelli *et al.,* "The European x-ray free-electron laser technical design report," DESY, Hamburg Report No. DESY 2006-097, 2007.

[4] Thales Group, 45 rue de Villiers, 92526 Neuilly-sur-Seine, Cedex, France www.thalesgroup.com

[5] W3C, "State Chart XML (SCXML): State Machine Notation for ControlAbstraction," http://www.w3.org/TR/scxml

[6] SCXML, "The Jakarta Project Commons SCXML," http://jakarta.apache.org/commons/scxml

[7] D. Harel, "Statecharts: A visual formalism for complex systems," *Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, June 1987.

[8] DOOCS, "DOOCS is a distributed control system," http://doocs.desy.de

[9] TINE, "Three-fold Integrated Networking Environment," http://tine.desy.de

[10] M. Penno *et al.,* "A Configurable Interlock System for RF Stations at XFEL," in *Proc. PCaPAC'08*, Ljubljana, Slovenia, Oct. 2008, paper WEZ03.

[11] Java Web Start, "Java Web Start Overview," http://www.oracle.com/technetwork/java/javase/overview-137531.html

[12] DOM, "Document Object Model Events," https://www.w3.org/TR/WD-DOM-Level-2/events.html

[13] UML, "Introduction to OMG's UML," http://www.uml.org/what-is-uml.htm

[14] Y. Renier *et al.,* "Fast Automatic Ramping of High Average Power Guns," in *Proc. IPAC'17*, Copenhagen, Denmark, May 2017. doi:10.18429/JACoW-IPAC2017-TUPIK052

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP25

EVOLUTION AND CONVERGENCE OF ALARM SYSTEMS IN TANGO

S.Rubio-Manrique, G.Cuni, F.Fernandez, R.Monge
ALBA Synchrotron, Cerdanyola del Vallès, Barcelona, Spain
G.Scalamera, Elettra-Sincrotrone, Basovizza, Trieste, Italy

Abstract

The technology upgrade that represents Tango 9 has triggered the evolution of two of the most used Tango tools, the PANIC Alarm System and the HDB++ Archiving. This paper presents the status of the collaboration between Alba and Elettra Synchrotron sources for the convergence of its both alarms systems under the IEC62682 [1] standard, and the usage of HDB++ tools for logging and diagnostic of alarms. Relevant use cases from the user point of view has been added to the paper as a validation of the benefits of this control system evolution.

INTRODUCTION

Alarms in Control Systems

Alarm Systems have been a common part of control system toolkits for decades. In the Synchrotron community some of the most common tools are PANIC [2] and AlarmHandler [3] for Tango Control System [4], as well as BEAST Alarm System for EPICS.

PANIC and AlarmHandler systems have coexisted within the Tango community for years, but at some point new members of the community asked whether to choose one or the other for their specific domain. This question triggered an effort to compare both systems, extract the best features of them and explore how they could complement each other.

This effort required from us to redefine what an Alarm System was, and what it was expected to do.

What's an Alarm System?

SKA and Elettra institutions proposed to the Tango community to adopt a common terminology and behaviour based on an international norm, the IEC 62682:2014 "Management of Alarm Systems for the Process Industries"[5-6]. The norm states that:

- The primary function of an Alarm System will be to notify abnormal process conditions or equipment malfunctions, and support the operator response.
- The Alarm System is NOT part of the protection nor safety systems, which must have separate tools following its own regulation.
- The Alarm System is part of Operator Response, thus it's part of the HMI (including the non-graphical part of it).

These three statements clarified what our Alarm Systems were expected to do; providing a common ground to start the collaboration on merging both projects.

Alarms within the Tango Control System

It is needed to introduce several concepts on how alarms are developed within a Tango Control System. The following terms describe the object hierarchy:

- Tango Host or Database: the central database where all devices are registered and configuration stored.
- Device Server: Each independent software process distributed across the system, managing one or several Tango devices.
- Device: a Tango Device is an entity that can be typically identified to a hardware piece or software process (e.g. a vacuum pump, a PLC, a motor, a voice synthesizer). Each device exports to the control system its Attributes (process variables), Commands (actions), Properties (for configuration) and States.
- Attributes: Each of the process variables exported by a Tango Device. They support both synchronous or asynchronous reading and writing. At each attribute reading it exports its value, timestamp and quality.
- Attribute Quality: The attribute quality accompanies each value to express the process conditions. Quality can be VALID, INVALID, WARNING or ALARM and it can be set on runtime by a Tango user specifying which ranges of operation/warning/alarm will trigger a quality change.

Quality-based vs Formula-based Alarms

The most primitive scope of Alarm Systems in Tango just included the logging of those attributes qualities in ALARM. But this approach didn't apply when it was required the interaction between multiple devices.

The Tango Alarm System (AlarmHandler device server and its alarm database) was developed [3] to enable logical and arithmetical operations on attribute values, thus extending the alarm triggering from the simple matching of an attribute value within valid ranges.

PANIC [7] was developed by ALBA Synchrotron in 2007 [8] as a Python [9] alternative to the Tango Alarm System. It was based on similar principles but applying a distributed architecture (see Fig. 1) and trying to add annunciator features and more flexibility [10] in alarms declaration, allowing to execute python code for both the formula and the resulting action [11]. This enabled the usage of wildcards for attribute selection, and reusing the data from the alarm evaluation to generate rich-text emails or SMS messaging.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP25

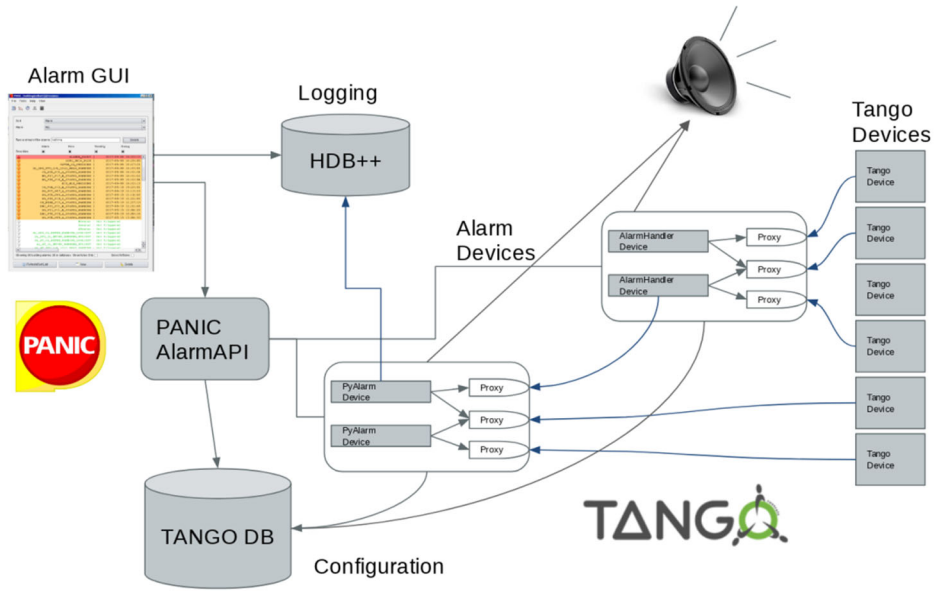


Figure 1: PANIC Architecture, showing alarm device servers, configuration and logging databases, and annunciators.

ALARM SYSTEMS CONVERGENCE

Existing Alarm Systems in Tango have been already described in previous papers [2-3], as well as the changes required [12] to adapt them to the IEC62682. This paper focus instead in the changes that allowed their convergence into a single toolkit.

Unifying the AlarmHander and PANIC device servers forced us to unify criteria on:

- Alarm Formula database
- Alarm attributes
- Classification of alarms
- Annunciators specification and triggering
- Priorities
- Exporting attributes to HMI
- Alarm Summaries

The criteria for convergence have been triggered by the chosen norm (that fixed terminology and state conditions), the Tango control system architecture (centralized database / distributed devices) and the possibilities to expand the HMI to the web (choosing JSON as the default data format for Alarm Summaries).

Intended to reduce the number of different tools required for having a functional alarm system, the Alarm database has been dropped to use Tango Properties instead. Keeping the alarm formulas and configuration in the Tango database helps to unify the quality-based and formula-based alarm approaches, thus storing the configuration for both types of alarms in the same database.

Following the same principle, it has been discarded the idea to maintain a separate Alarm logging, thus reusing the new HDB++ archiving for Tango [13].

HDB++ archiving allows to record alarms on state change, and at the same time records all the events received by the attributes involved in the formula (including both value and quality changes). Thus, alarm changes and attribute changes can be queried and represented from the same database using already existing tools.

ALARMS IMPLEMENTATION

Reactive Alarms vs Polled Alarms

The PANIC API is currently connecting to two types of device servers, PyAlarm, developed in Python by ALBA Synchrotron, and AlarmHandler, developed in C++ by Elettra Sincrotrone.

Both device servers acquire lists of formulas from the Tango Database Properties, connect to the attributes appearing on the formulas and evaluate the results on each attribute value change; triggering actions when required by an alarm state change. Devices differ on how they connect to attributes and react to value change.

PyAlarm is polled-based, so it means that attributes are read periodically, updating the formula result at each attribute reading. Attribute changes are cached and alarm is not triggered until the result is True for a number of iterations (the AlarmThreshold property). With a threshold of 1, response time can be as short as 100 ms; but still this minimal latency time is needed. In exchange, complex formulas are enabled and the polling buffer allows to use the average, the max peaks or the delta change of the last N values acquired.

AlarmHandler instead is event-based. It means that the device is just waiting for events sent from the Tango Control System, reacting immediately when a change event is received. In this case, latency is minimal, and actions are executed immediately. Alarm formula parsing is done in C++, and it is restricted to basic arithmetical and logical operations between attribute values.

As demonstrated in the field, both systems are complementary, as the AlarmHandler can be dedicated to fast reaction on critical conditions while PyAlarm flexibility

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP25

provides its best usability on interacting with multiple devices or evaluating attribute evolution in time.

Alarm System Scalability

Another factor that triggered convergence of Alarm Systems in Tango has been the need to increase system scalability and performance for the new SKA project.

Alarms Systems can be scaled using different approaches. PANIC allows scaling by exporting each evaluated alarm as a new attribute. Thus, new alarms can be written using the result of the previous ones already evaluated. It allows having a detailed alarm for each subsystem of a sector, and then summarize all values in a single sector alarm. As example, vacuum alarms will contribute to the alarm state of a sector, and at the same time are grouped in a vacuum alarm for all sectors.

Alarm summaries can be done at client level (Alarm View) or at device server level (Alarm Group). When created at alarm level, it will behave as any other alarm, triggering actions and exporting a new attribute, so the system can be scaled to the next level. They can be accessed even from other Tango Control Systems, so the hierarchy can be expanded indefinitely. Propagation times at each level can be tuned using the AlarmThreshold and PollingPeriod properties.

Alarm Annunciators

Besides its initial features (email, SMS, logging), the current version of PANIC allows now to trigger any kind of Tango command or python code script and provides the capability of reusing alarm data and values in the execution of these commands. As example, it allows to send alarm description to speakers, include attribute values in an email or html report, move a motor a fixed number of steps depending on a calculation, etc.

It also expands the logging mechanisms. HDB++ and the Tango DB are used to keep history of alarm and attribute changes and its configuration; but via external commands other messaging or logging systems can be easily integrated. It allowed PANIC to interact with some popular applications like Telegram or Kibana and generate web reports based on JSON files [14].

CONCLUSIONS AND FUTURE

PANIC is a key tool for the daily operation of ALBA Synchrotron Accelerators and Beamlines, it is the first diagnostic tool used in order to check any incidence. It offers an overview of currently triggered alarms, allowing a full comprehension of the problem. Its second potential resides in the customization of the alarms applications that allows to suit the specific needs of each user group and, in its integration with Taurus [15-16], eases a fast response and troubleshooting in a user-friendly environment.

Having a common interface with Elettra's AlarmHandler allow to expand PANIC functionality to a new level, as now two complementary Alarm engines are provided and scalability can be achieved combining performance and flexibility at each level.

PANIC Use Cases

This are some practical examples of PANIC use cases that have been enabled by the changes introduced in the latest releases:

Protection of infrared mirrors; in this case an alarm was required to, in case of a fast increase of temperature, to be able to move a mirror outside of the vacuum chamber in less than 200 ms. For this application alarms on delta change, response to events and precise motor movement execution were required.

Injection Permits: In the last upgrades of our Synchrotron facility, the number of injection modes are increasing as well as the complexity of the rules that allow to inject in the storage ring. PANIC has been used as an easy tool to compile permission rules and convert it into attributes that can be used as enable/disable of different operations.

Vacuum Systems: it is very important the continuous monitoring of the pressure and equipment state, but besides this, there are many variables belonging to other sub-systems which could affect the vacuum status. PANIC allows to extend the views and notifications of each alarm to include additional information or parameters that help to diagnose the problem. It also allows to crosscheck temperatures against different ranges depending on the current operation status (injection, maintenance, bakeout); checking not only the value but its proper behaviour.

Testing and Deployment

Testing of PANIC is critical in to ensure its reliability and scalability, guaranteeing that any modification in the evaluation engine does not break the compatibility with previous versions. A test suite has been developed [17] and its currently expanded with the help of tango-simlib project. Testing will be introduced in our continuous integration/deployment cycle based on Debian packages.

ACKNOWLEDGEMENT

The latest release of PANIC contains contributions from many people: MaxIV, ESRF, ALBA, Elettra and Solaris institutes as well as some private companies like S2Innovation, IK and TCS.

The work of validating and suffering each new release is shared with our users, the Operation and Vacuum groups at ALBA. They contributed with their experience to refine and improve alarm formulas and syntax.

REFERENCES

[1] *Management of alarms systems for the process industries,* IEC-62682, IEC, 2014.

[2] S. Rubio-Manrique et al., "PANIC, A Suite for Visualization, Logging and Notification of Incidents.", in *Proc. PcaPAC'14*, Karlsruhe, Germany, Oct. 2014, paper FCO206.

[3] L. Pivetta, "Development of the Tango Alarm System", in *Proc. ICALEPCS'05*, Geneva, Switzerland, Oct. 2005, paper WE3b.1-70.

[4] TANGO, http://www.tango-controls.org

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP25

[5] *Engineering Equipment and Material,* Users'Association (EEMUA) issued publication, 191, University of Manchester, United Kingdom, 1999.

[6] M. Tennant, "Implementing Alarm Management Per the ANSI/ISA-18.2 Standard", *Control Engineering*, September 2013.

[7] PANIC, https://github.com/tango-controls/panic

[8] S. Rubio-Manrique et al. "Extending Alarm Handling in Tango", in *Proc. ICALEPCS'11*, Grenoble, France, Oct. 2011, paper MOMMU001.

[9] D. Fernández et al. "Alba, a Tango based Control System in Python", in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, paper WEPMU005.

[10] S. Rubio et al., "Dynamic Attributes and other functional flexibilities of PyTango", in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, paper THP079.

[11] http://www.pythonhosted.org/panic/recipes.html

[12] G. Scalamera, L.Pivetta, S.Rubio-Manrique, "New developments for the Tango Alarm System", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, paper TUPHA165.

[13] L. Pivetta et al., "New developments for the HDB++ Tango Archiving System", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, paper TUPHA 166.

[14] M. Broseta, D. Roldan, S. Rubio, A. Burgos, G. Cuni, "A web-based report tool for Tango Control Systems via websockets", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, paper TUPHA 173.

[15] S. Rubio et al., "Unifying all Tango Services in a single Control Application", in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015. paper WEPGF148.

[16] Taurus Library, http://www.taurus-scada.org

[17] S. Rubio-Manrique et al., "Reproduce Anything, Anywhere: A Generic Simulation Suite for Tango Control Systems", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, paper TUDPL01.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP26

THE DEVELOPMENT OF VACUUM GAUGE MONITORING AND CONTROL SYSTEM USING Web2cToolkit

H. Ishii †, T. Kosuge, H. Nitani
High Energy Accelerator Research Organization(KEK), 1-1 Oho, Tsukuba, Ibaraki 305-0801, Japan

Abstract

The Photon Factory is an accelerator-based light source facility that is a part of the High Energy Accelerator Research Organization (KEK) in Japan. At the Photon Factory, a message transferring software named Simple Transmission and Retrieval System (STARS) is used for beamline control. STARS is suitable for small scale remote-control systems using TCP/IP sockets and works with various types of operating systems.

STARS is applicable to various control systems; we developed the vacuum gauge monitoring and control system with STARS in this work.

Web2cToolkit is a framework developed by Deutsches Elektronen Synchrotron (DESY) that provides a user-friendly human machine interface, and developers can easily create web-based graphical user interfaces (GUIs) with Web2cToolkit.

Web2cToolkit supports various types of protocols (e.g., TINE, DOOCS, EPICS, TANGO), including the STARS protocol. We adopt Web2cToolkit as a framework for a front-end GUI application of our vacuum gauge monitoring and control system. Although the application development is still in progress, some features have already been implemented.

OVERVIEW OF STARS

Simple transmission and retrieval system (STARS) [1] is composed of at least one server program "STARS server" and one client program "STARS client". Each STARS client is connected to the STARS server through a unique node name and handles text-based messages through a TCP/IP socket (Fig. 1).

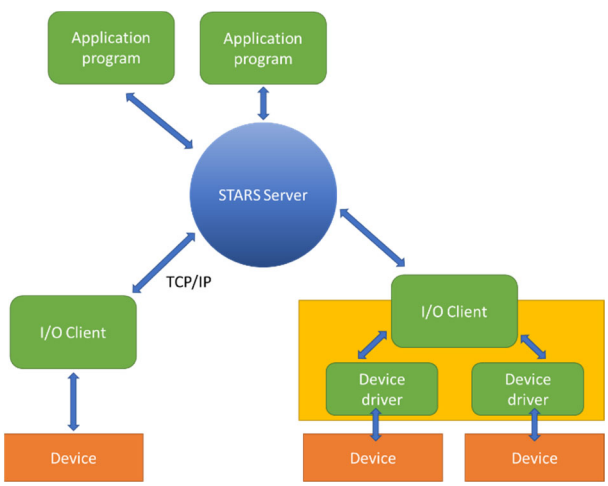


Figure 1: Communication between STARS clients and a STARS server.

The STARS server program is written in Perl. Therefore, it can be executed on different operating systems, such as Windows, MacOS, and Linux. STARS has been introduced as a beamline control system in more than 20 beamlines of the Photon Factory. In addition, an interface program of the beamline interlock systems has been developed using the STARS software.

VACUUM GAUGE CONTROLLER

A large number of vacuum gauges, exceeding 300, are installed in the beamlines at the Photon Factory. These vacuum gauges are connected to various types of controllers that monitor the degree of vacuum of the beamlines. Each controller has a relay output port that either completes or disrupts an electric circuit according to the degree of vacuum, and this port is used for the beamline interlock system. The vacuum environment of the beamlines at the Photon Factory is managed by this beamline interlock system (Fig. 2). The thresholds for these relays can be configured using the switches on the front panels of the controllers or through their respective computer interfaces.

Most of the vacuum controllers have computer interfaces for monitoring the degree of vacuum and operating the controller. Data can be obtained or set through these interfaces and include the following:

- Getting the current value of vacuum;
- Turning the sensor device on or off;
- Setting and Getting the thresholds of the relay output function.

The interface of each vacuum controller has similar functions; however, the different types of communication protocols depend on hardware type or manufacturer.

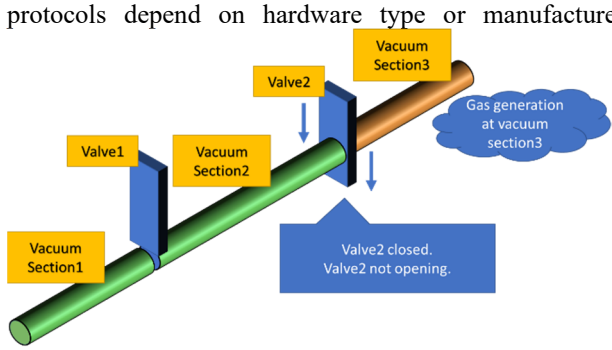


Figure 2: Beamline interlock system and vacuum monitoring.

VACUUM MONITORING SYSTEM

The beamline interlock system uses only the relay output functions; therefore, it is impossible to protect the beamline before a vacuum hazard occurs. The vacuum hazard will interrupt all experiments via leaked synchrotron

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP26

radiation, so it is important to detect abnormal situations even if is the changes may be insignificant. A vacuum data monitoring system has many benefits, such as prediction of vacuum-related issues of the beamline components.

Therefore, we developed the vacuum monitoring and data logging system shown in Fig. 3. We installed STARS on the vacuum gauge monitoring system, and all vacuum gauge controllers were connected to the STARS server through a TCP/IP socket with a serial interface device server.

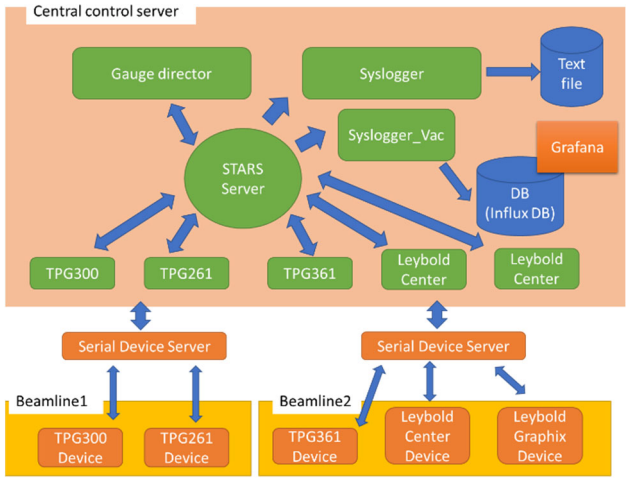


Figure 3: Vacuum monitoring system.

To transmit and receive messages between each vacuum gauge controller and the STARS server, a client soft-ware was developed, and the following client program was installed on each vacuum gauge controller (Table 1). Each client software instance has a unique identifier that consists of the beamline and gauge controller names (e.g., "bl1ga1," where "bl1" is the beamline name and "ga1" is the gauge controller name).

Table 1: List of Vacuum Gauge Controllers Used in Photon Factory

Controller Type	Computer Interface	Manufacturer
TPG300	RS-232C	Pfeiffer Vacuum, GmbH
TPG261	RS-232C	Pfeiffer Vacuum, GmbH
TPG361/362	USB	Pfeiffer Vacuum, GmbH
Canter series	RS-232C	Leybold Vacuum, GmbH
Graphix series	RS-232C	Leybold Vacuum, GmbH

DEVELOPMENT OF SOFTWARE

We developed individual STARS client software for the gauge monitoring system, and each client software has different functions, which are described below.

Vacuum Gauge Control Client

Typically, the original commands of the vacuum gauge controller are not compatible for another type of controller. This is inconvenient for developers because the soft-ware requires modification in accordance with the type of controller. To resolve this problem, the client software has command converters between the original commands and the unified commands, which are defined for other STARS client softwares. For example, some common unified commands are as follows: "Get-Value," "SetSFunction," "GetSwFunction," "SPS," and so on. Using these commands, we can obtain information such as vacuum data, device setting data, and device condition.

Gauge Director Client and Data Logger Client

A gauge director client automatically sends commands according to the configuration file, similar to the Cron utility in a Unix system. The configuration parameters are the sending interval, transmission command, and target node name. The gauge director client sends the "GetValue" command to all vacuum gauge control clients every 30 s in our vacuum monitoring system. The data logger client creates a log file for all STARS communication messages, including vacuum information, and sends the log data to an external database server. The log file is a text-based archive, and InfluxDB is used as the external database system.

Graphical Interface

Grafana is an open source software for data visualization. We adapted Grafana to visualize vacuum logging data, which are recorded in the InfluxDB server. The log viewing interface is called "Dashboard" and can be easily edited on using a web-browser. Grafana has different kinds of graphical components such as graphs, tables, and so on. We used a graph component to display the time variation of the degree of vacuum in the beamlines (Fig. 4).

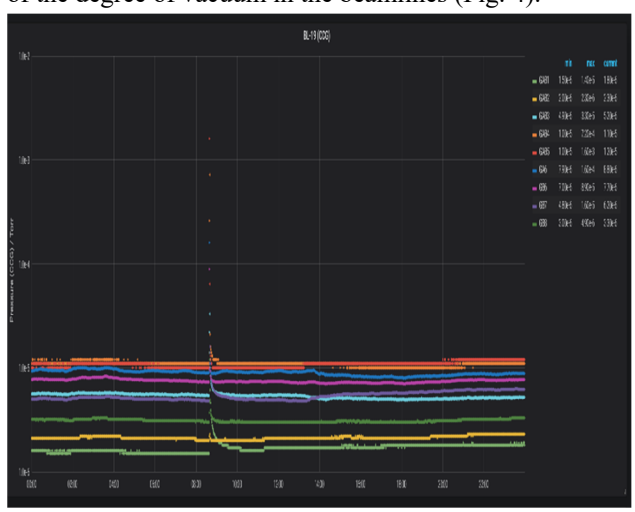


Figure 4: Example of "Dashboard" created with Grafana.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP26

STARS FOR WEB2C

The Web2c viewer is a configurable, browser based, dynamic, and bidirectional internet application for live displays (Fig. 5).

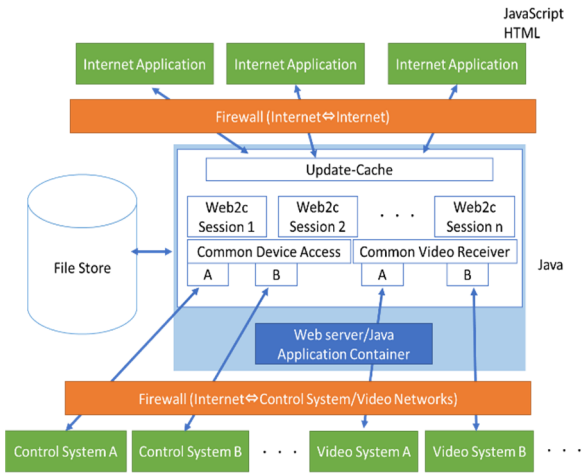


Figure 5: Web2c viewer.

Each application page is easily created using the Web2c viewer wizard or by directly editing the XML file. We created the application page to obtain the information from the beamline vacuum gauge controller using Web2c viewer wizard. The application page consists of various components listed in a corresponding configuration file [2]. The properties of the components are tooltip text, font size, arrangement, signal source URI, and so on. We used the "Web2cViewerText" viewer component in this case (Fig. 6). Currently, only a few Web2c viewer components are available on STARS. For example, "Web2cViewerButton, " "Web2cViewerLabel, " and "Web2cViewerText."

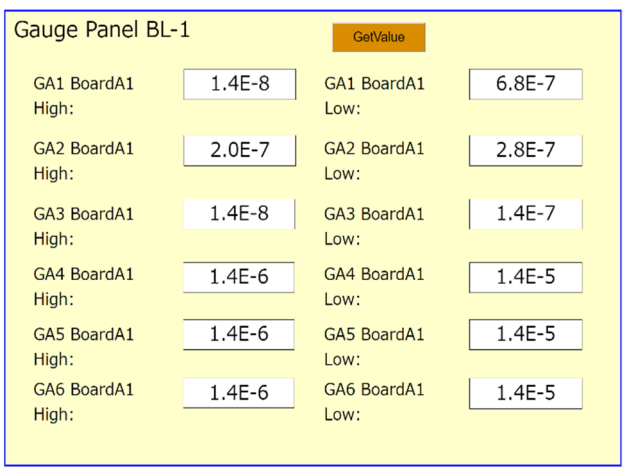


Figure 6: Web2c application page for the vacuum gauge controller.

In Fig. 6, the threshold values of the relay function of the vacuum gauge controller, which is obtained through STARS, are displayed in the text box. When the "GetValue" button is pressed, the threshold values are updated to present values. The Web2c application communicates with STARS via the "STARS proxy." The STARS proxy is the communication protocol between Web2c and the STARS client, and it converts the Web2c message to the STARS message of the gauge controller client. Both Web2c application and STARS proxy are connected to the STARS server as a STARS client (Fig. 7).

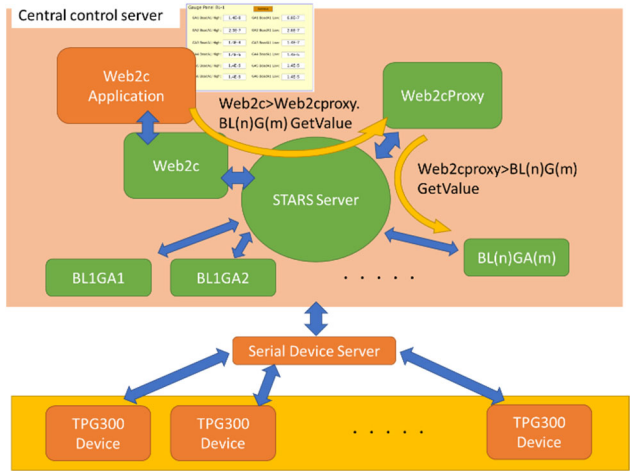


Figure 7: STARS server for Web2c.

CONCLUSION

We constructed a vacuum monitoring system in conjunction with STARS and Web2c. This system supports various types of vacuum gauge controllers. These vacuum gauge controllers connect to the STARS server through a TCP/IP socket and a have unified command system. The vacuum data thus recorded can be displayed using graphs on a web browser. Web2c is a browser-based internet application that can obtain the threshold of a vacuum gauge controller through STARS. We are refining the STARS client program, which is associated with the Web2c framework, and intend to advance the development of other Web2c viewer components through STARS.

REFERENCES

[1] STARS, http://stars.kek.jp
[2] Web2c Toolkit Framework for Web-based Controls Clients, http://adweb.desy.de/mcs/web2cToolkit/web2c_home.htm

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP27

DEVELOPMENT OF THE MALFUNCTIONS DETECTION SYSTEM AT VEPP-2000

O. S. Shubina†, A. I. Senchenko
BINP SB RAS and Novosibirsk State University, Novosibirsk, Russia

Abstract

In 2007, the creation of the electron-positron collider VEPP-2000 (Fig. 1) was completed at the Institute of Nuclear Physics of the SB RAS. The VEPP-2000 collider facility consists of various subsystems, and a failure of any subsystem can lead to the incorrect operation of the complex for several hours or even days. Thus, there is a need to create software that will warn about possible malfunctions. To accomplish the task, software was developed consisting of three modules. The first performs automatic verification of compliance data obtained from the accelerator complex and rules describing the correct subsystems operation. The second module is a user-friendly web interface that displays information about the state of the complex in a convenient way. The third module acts as some intermediary between the first and the second. It processes messages arriving at the message queue and redirects them to all subscribed clients via the web socket. This article is devoted to the development of test software, that is currently running on the VEPP-2000 control panel.

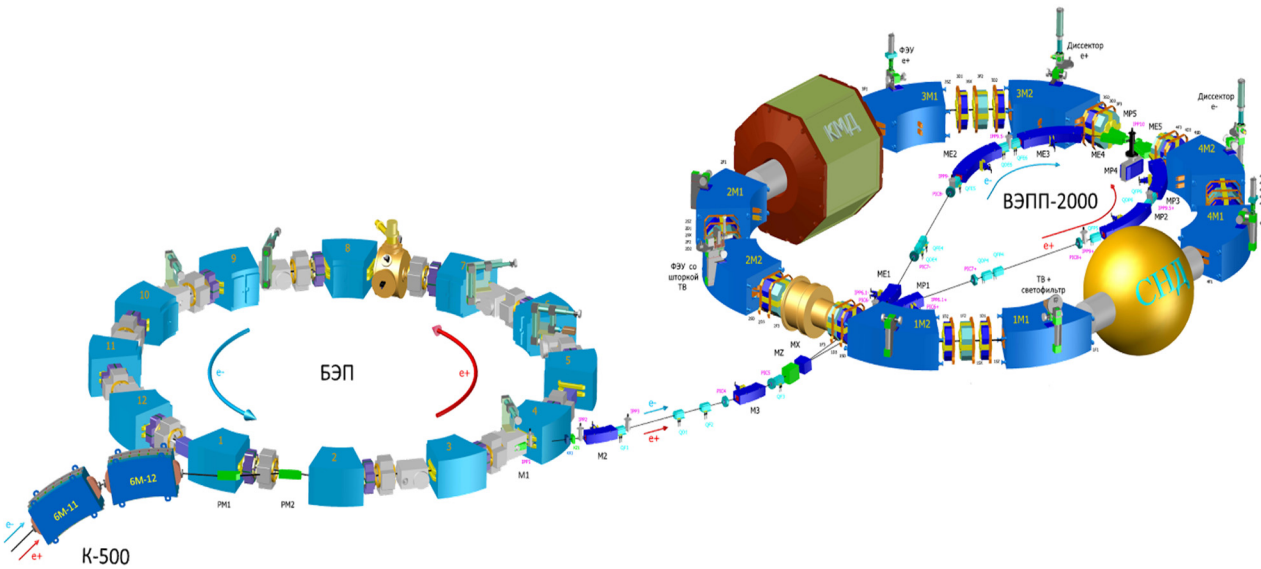


Figure 1: VEPP-2000.

INTRODUCTION

The VEPP 2000 is an electron-positron collider, that was commissioned at the Budker Institute of Nuclear Physics. The VEPP-2000 acceleration complex consists of a few main subsystems: BEP booster ring and VEPP-2000 collider ring. Beam diagnostics is based on 16 optical CCD cameras that register the synchrotron light from either end of the bending magnets and give the full information about beam positions, intensities and profiles. In addition to optical BPMs [1], there are also 4 pick-up stations in the technical straight sections and one current transformer as an absolute current meter. The VEPP-2000 control system [2] has a complex structure and consists of about 4000 channels for monitoring and control. This complicates the timely detection of incorrect operation of the complex or any malfunctions. To solve this problem, software was developed consisting of various parts. The core of the software is a troubleshooting module, that performs automatic validation of rules stored in special configuration files. The following component is responsible for delivering information about the status of the complex to subscribers via the web sockets. Also for the timely notification of new subscribers about the state of the complex uses caching of the state snapshot in the Redis database. Finally, the graphical interface provides a convenient view of all the necessary information.

MODULE FOR CHECKING THE STATE OF THE COMPLEX

The hardware part of the VEPP-2000 accelerator complex has a complicated architecture; therefore, a group system has been developed for the convenience of troubleshooting. Each group is a collection of several channels or

† olgashubina2011@gmail.com.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP27

subsystems of the complex, united according to some principle. Thus, during processing, the system operates directly with these groups, rather than with individual channels. Information about groups is stored in a special configuration file. After initialization of all groups and obtaining the necessary data (point 2), the process of troubleshooting is started in accordance with the rules specified in the configuration file. First of all, the system creates a set of objects corresponding to each individual group and storing the information about the rule that is applied to this group, critical and boundary values and a reference to the corresponding data buffer. Each object is an instance of a specific class in which these or other troubleshooting rules are implemented. Then, with a periodicity, a cyclical group check is performed for the presence of malfunction, as well as for the activity of the data source. After the check, a report on the complex status is generated and sent to the message broker RabbitMQ.

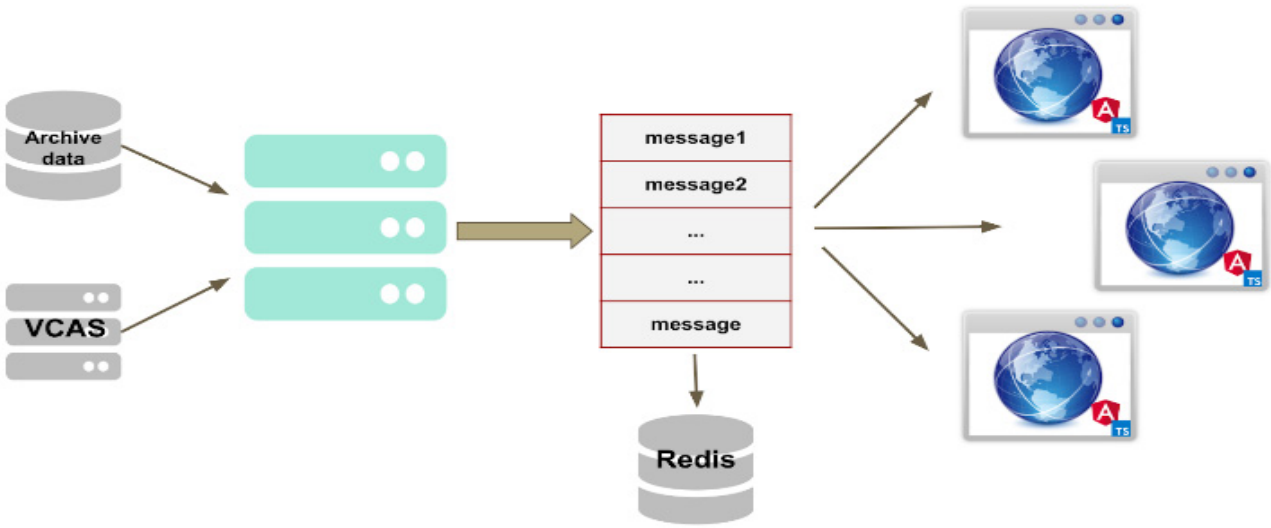


Figure 2: General scheme.

CREATE AND UPDATE DATA BUFFER

Before starting the troubleshooting process, a data buffer is created in the system. Data enters the buffer from the database through a specially developed API for interacting with the accelerator complex archive data. The buffer is filled with data on all the necessary channels for a certain period of time, that is specified in the configuration file groups.

After creating the buffer, it starts the automatic update process. The system subscribes to the VCAS server to obtain online data. Data storage is organized in the form of a ring buffer, i.e. each time, when new data is received, they are saved to the buffer, and the data that is outdated is deleted from the buffer.

The main problem that arose when designing this software is the use of memory. There was a need to create a common memory buffer with which two processes will work in parallel: automatic buffer update and troubleshooting process. At the same time, the solution had to have good performance and be easily incorporated into the developed software and the management system as a whole. This problem was solved by queuing in distributed memory from the multiprocessing library of Python.

MESSAGE BROKER AND COMPLEX STATUS SNAPSHOT

For the distribution of information between customers, a special service, developed on the basis of the message broker RabbitMQ [3], is responsible. It is a web socket server. Any user or software can subscribe this server and receive information on the state of the complex with a certain periodicity.

After passing the next check iteration, a report on the state of the complex is generated and sent to the message broker. Then, the report is processed and sent to all existing subscribers. To reduce traffic and the load on the system, information about the state of the complex is not sent at every iteration of the test, but only when the state changes. Therefore, there may be a problem that the new subscriber does not receive information about the state of the complex for a long time.

To solve this problem, it was decided not only to transmit information online, but also to save the state cast in-memory data structure store Redis. The storage scheme of the state cast has the structure shown in Fig. 2. And now, when adding a new subscription, it will read up-to-date information from the repository, and then receive updates via the web socket when the state changes.

Each author should submit the PDF file and all source files (text and figures) to enable the paper to be reconstructed if there are processing difficulties

MALFUNCTION DETECTION METHODS

Currently, several methods of troubleshooting have been developed to test this system (Fig. 3), in part for current

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP27

and vacuum systems. To troubleshoot current systems, three simple rules are used:

- check of the equality of the set current to the specified parameter
- check of the difference between the set and obtained value
- the standard deviation of the normalized value

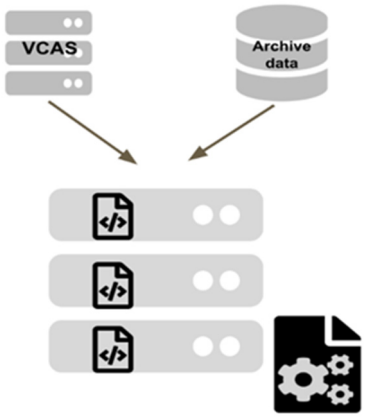


Figure 3: Malfunction detection scheme.

The second fault finding algorithm was developed for those subsystems in which measurements are rarely made to detect a malfunction in time. And in this case, the algorithm performs:

- interpolation of values by the most optimal function
- extrapolating values for current or future time

Templates are provided for recommended software and authors are advised to use them. Please consult the individual conference help pages if questions arise.

WEB GUI

For the convenience of displaying information about the state of the complex, a graphical web interface (Fig. 4) was developed based on the Angular 5 framework [4]. The first consists of colored indicators of each group. Here, red means that an error has occurred on this subsystem, yellow indicates a situation close to critical, green indicates a correct operation. Also, if the data source for any of the subsystems is unavailable, then it is displayed in gray. In addition to the color indication, there is a module in that an informational error message is displayed indicating the time of the error and the subsystem on that it occurred.

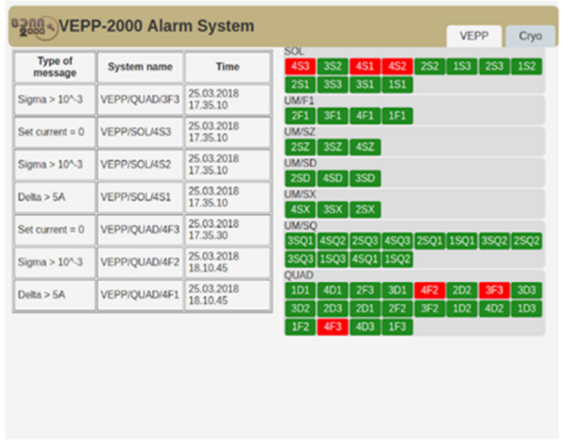


Figure 4: WEB GUI.

CONCLUSION

At present, a prototype of a troubleshooting system at the pulping VEPP-2000 has been implemented and launched in test mode. As part of this work, several algorithms for checking the state of the complex were implemented and connected to an automatic check. In the future we plan to expand the set of rules, as well as review the way they are stored. In addition, the task is to improve the performance of this system, and solve a number of problems associated with distributed memory and timely notification of the state of the complex. It is also planned to expand the scope of their research not only as applied to troubleshooting, but also to their automatic prevention.

REFERENCES

[1] Yu. Shatunov *et al.*, "Project of a New ElectronPositron Collider VEPP-2000", in *Proc. EPAC'00*, Vienna, Austria, p.439.

[2] A. Senchenko *et al.*, "VEPP2000 Collider Control System", in *Proc. PCaPAC'12*, Kolkata, India, Dec. 2012, paper FRCB04.

[3] RabbitMQ official site, `https://www.rabbitmq.com`

[4] Angular5 official site, `https://angular.io`

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP28

DEVELOPMENT OF SOFTWARE FOR ACCESSING THE VEPP-2000 COLLIDER FACILITY ARCHIVING SYSTEM

O. S. Shubina†, A.I.Senchenko, P.Yu. Shatunov
BINP SB RAS and Novosibirsk State University, Novosibirsk, Russia

Abstract

The VEPP-2000 is an electron-positron collider, that was commissioned at Budker Institute of Nuclear Physics. The VEPP-2000 acceleration complex consists of a few main subsystems: BEP booster ring and VEPP-2000 collider ring. Data from accelerator complex are recorded regularly with a frequency at 1 Hz. There is often a need to obtain already stored data for analysis or modeling. In addition, you must provide remote data access to optimize the workflow. The solution of this problem must be universal, and it must be easily adapted to various databases and installation modes. To solve the task, the software was developed based on the client-server architecture. The server part is responsible for processing data in automatic mode according to the developed algorithm. The client part allows to view the data in a user-friendly form. This article talks about the development of software, simplifying access to the VEPP-2000 archiving system, that is launched on the VEPP-2000 control panel.

INTRODUCTION

The VEPP 2000 is an electron-positron collider located at the Budker Institute of Nuclear Physics. The VEPP-2000 acceleration complex (Fig. 1) consists of two main subsystems: the BEP booster ring and the VEPP-2000 collider ring. Beam diagnostics system is based on 16 optical CCD cameras that register the synchrotron light from both ends of the bending magnets and provide full information about beam positions, intensities and profiles. In addition to optical BPMs [1], there are also 4 pick-up stations in the technical straight sections and the one current transformer working as an absolute current meter.

Over than 1200 control channels and 2400 monitoring channels and their joint usage impose rigid restriction on the control system [2]. The architecture of VEPP-2000 software is based on traditional three-layer structure. The important application in the middleware layer is an elaborately designed Log Server [3]. Its purpose is to store all necessary automation system data to the storage.

Every day various accelerator-based experiments are carried out. Sometimes researchers need to obtain the past data for the analysis and calculations. Moreover, it will be useful to have remote access to this data.

Thus, there is the problem of access to the data stored on the hard disk. Also, the solution of this problem have to be universal, and easily adapted to various databases and installation modes.

THE GENERAL SCHEME OF THE APPLICATION

To solve the task, the software has been developed based on the client-server architecture (Fig. 2). HTTP has been chosen as a transport layer protocol, since it is the most common method of data transmission. And also, the REST approach has formed the basis of this application.

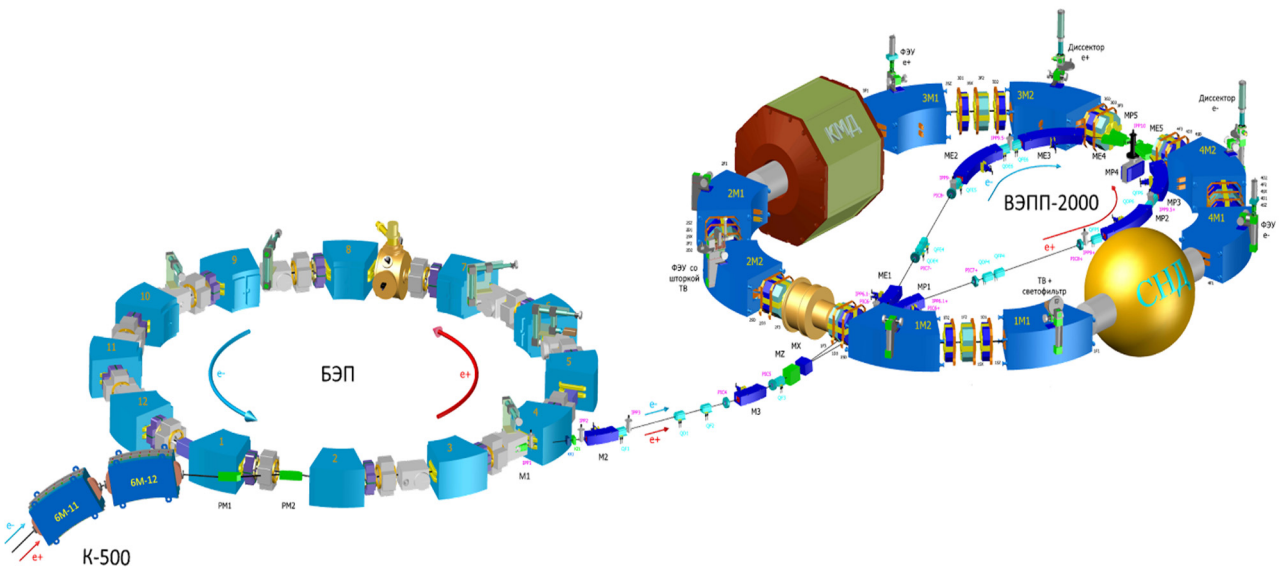


Figure 1: VEPP-2000.

† olgashubina2011@gmail.com.

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP28

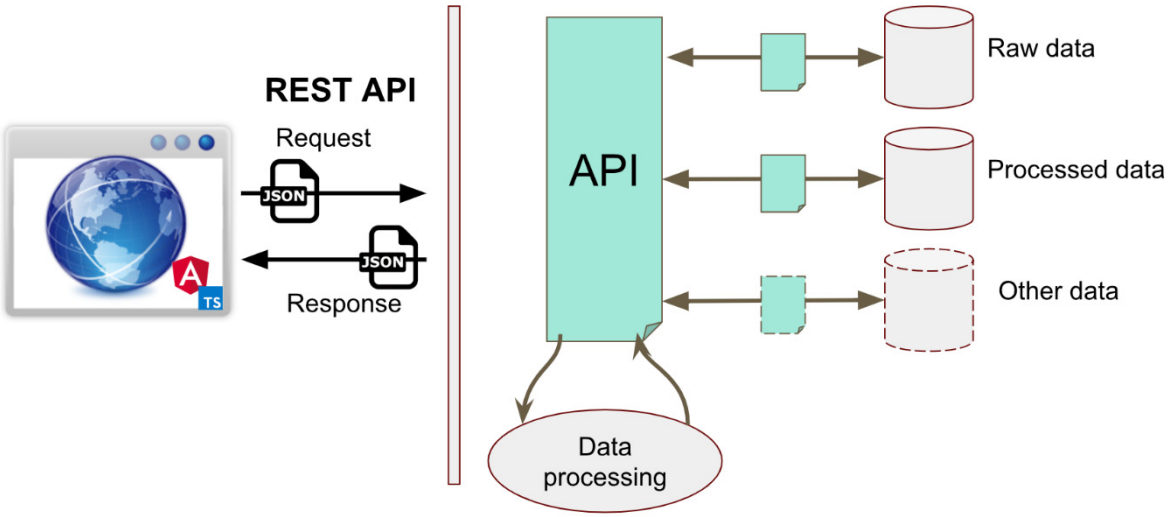


Figure 2: General scheme.

The interaction between the client and the server is carried out using standard HTTP methods. When the user enters all necessary data, the client part of the application generates a special request and passes it as a JSON format message, which has the following structure:

- t1 - the beginning of the requested time interval;
- t2 - the end of the requested time interval;
- channels - the list of channels;
- fields - the method of data processing;
- level - the level of compression.

Next, the server processes the request and sends the response also in JSON format. Then the client works on the received data and provides it to the user. Below we will consider in more detail how the client and server parts of the software are arranged.

THE SERVER PART OF THE APPLICATION

To create the server part, the Flask web framework was used in the Python programming language using the Jinja2 template engine. This framework was chosen because it is designed to create small web applications that have basic capabilities, which are sufficient for our system.

The server part (Fig. 3) provides a single-entry point for obtaining the necessary data via the URL identifier. Its main task is to handle requests coming from the client and to issue data that satisfy the query parameters. Rest approach allows to hide from the client the implementation of specific procedures describing the structure of data storage, as well as procedures for interacting with it.

The server has several main components. The first component is responsible for provision of data from the database. Since it was necessary to create the most universal tool for interacting with data, it was decided to implement a unified software interface for accessing data from various sources. It allows to operate data through small interfaces that are implemented for a specific storage source.

The VEPP-2000 archiving system stores all the necessary data in the object-relational database of PostgreSQL. For example, for a month, the amount of data stored is about 900 million records, which is equivalent to 62 GB of disk usage. Thus, there is the problem of convenient representation, a large amount of data for quick viewing. Therefore, the component responsible for processing the data according to the developed algorithms was created. Since, amount of raw data is growing, there is a need to average new data regularly. To solve this problem, the Celery [4] task manager was used.

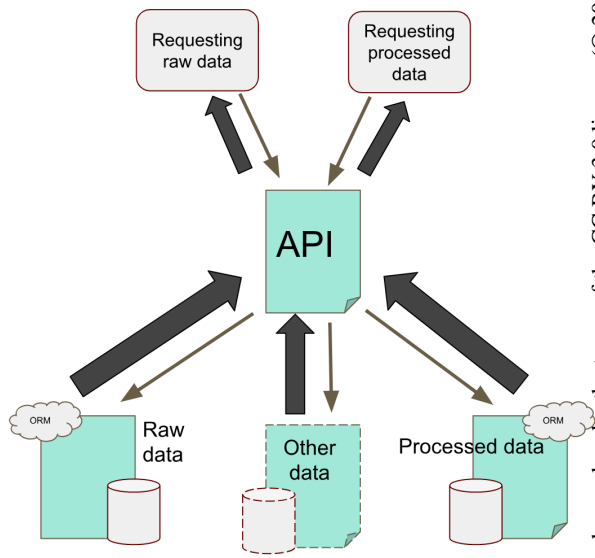


Figure 3: Server part scheme.

In the VEPP-2000 control system, currently, there are two types of channels: the first accepts text values, the second accepts a floating-point number. Depending on the type of data received, two data processing scenarios were developed. For numerical values, this is a finding for a certain period of time:

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP28

- arithmetic mean;
- the minimum value;
- the maximum value;
- the median value;
- root mean square deviation.

And for text:

- the left boundary value;
- right boundary value;
- the most significant value;
- central importance.

The last method processes data with both textual and numerical values. Data is processed by the methods described above at three different intervals. Thus, we obtain a group of tables consisting of two processing methods for three levels of compression.

THE CLIENT PART OF THE APPLICATION

The next stage of our work was the creation of the client part. The graphical interface (Fig. 4) was developed on the basis of the Angular5 [5] framework using the TypeScript language.

The main tasks of the GUI are the providing a convenient viewing of channels with various details, and the providing an interface to access data. For the convenience, the channel list was organized as a drop-down tree, similar to a hierarchical file system. For example, the path to the VEPP / Currents / FZ channel will look like this:

- VEPP
  - Currents
    - VEPP / Currents / FZ

Since only one selected value is drawn when displaying the processed data graph, for viewing convenience, when hovering to a specific point, the system displays the remaining values calculated for the given time.

Also, for a more comfortable viewing of the resulting graphs it is possible to scale the graphics and switch between different levels of compression. And also, it provides a convenient navigation on the graphic of mouse movement.

Since data from channels that have textual values cannot be represented in a graph similar to a graph with numerical values, it was customary to visualize information with text values using a table. This method is a kind of graphical interpretation of the tables stored in the database. Also, for more convenience, in the form of a table, it's possible to display data with numerical values.

Despite the fact that the graphical interface was faced with the task of providing fast and comfortable viewing of data, sometimes there might be a need to get raw data for a long period. In this case, the application provides a system for generating a script in Python. It has the same interface as for receiving data, only thus, the user is given a script displayed, which, also, is available for download. This script, when it is launched, connects to the server and uploads the data to a file located on the computer from which it is running.

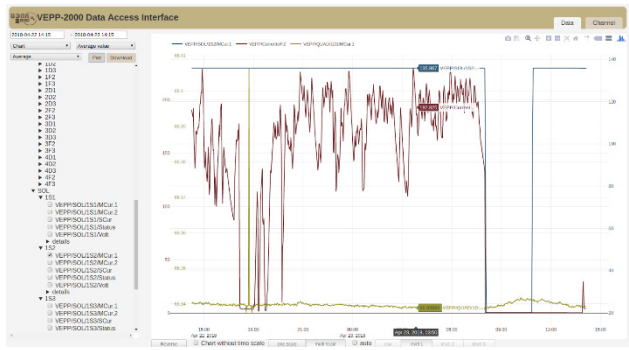


Figure 4: Web GUI.

CONCLUSION

As a result of this work, the software, that provides remote access to the data of the archiving system of the VEPP-2000 accelerator complex, has been developed. Using the REST approach provided a single point of access to data from various sources. As a result, a sufficiently flexible system was obtained in which the client's implementation does not depend on the implementation of the server, which allows, if it is necessary, to make minor or even significant changes to one of the parts without affecting the other at all.

For the convenience of viewing large amounts of data, the algorithms for processing them were implemented. According to the conducted tests, these algorithms allowed to increase the speed of data acquisition on about 90 times. Since the raw data will be regularly replenished, a task manager has been created that performs automatic data processing, according to the developed algorithms. Also, the approach, that allows to avoid the problems of reconnection, has been implemented.

REFERENCES

[1] Yu. Shatunov *et al.*, "Project of a New ElectronPositron Collider VEPP-2000", in *Proc. EPAC'00*, Vienna, Austria, p. 439.

[2] A. Senchenko *et al.*, "VEPP2000 Collider Control System", in *Proc. PCaPAC'12*, Kolkata, India, Dec. 2012, paper FRCB04.

[3] A. Senchenko, D. Berkaev, "VEPP2000 Logging System", in *Proc. PCaPAC'12*, Kolkata, India, Dec. 2012, paper WEPD14.

[4] Celery official site, http://www.celeryproject.org

[5] Angular5 official site, https://angular.io

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP30

DESIGN OF RELIABLE CONTROL WITH STAR-TOPOLOGY FIELDBUS COMMUNICATION FOR AN ELECTRON CYCLOTRON RESONANCE ION SOURCE AT RIBF

A. Uchiyama†, T. Nagatomo, Y. Higurashi, J. Ohnishi, T. Nakagawa, M. Komiyama, N. Fukunishi
RIKEN Nishina Center, Wako 351-0198, Japan
H. Yamauchi, M. Tamura, K. Kaneko, SHI Accelerator Service, Ltd., Tokyo 141-0032, Japan

Abstract

In the RIKEN Radioactive Isotope Beam Factory (RIBF) project, a superconducting linear accelerator has been implemented to enhance the beam energy necessary for promoting super-heavy element search experiments. A new 28-GHz electron cyclotron resonance ion source (ECRIS) has been installed upstream of it. Its control system has been planned to comprise the Yokogawa FA-M3V series, which is a programmable logic controller (PLC) with Experimental Physics and Industrial Control System (EPICS) because basically the same control system has been successfully operated for our existing ECRIS control system. However, the existing ECRIS control system with PLCs has a disadvantage of low reliability for communications between PLC stations. In addition, higher expandability is required because some devices, such as a power supply for an oven, will be changed depending on ion species produced at the ion source. In the new system, we have designed the control system by utilizing a star-topology fieldbus for communications between the PLC stations to establish safety and expandability.

INTRODUCTION

The RIKEN Radioactive Isotope Beam Factory (RIBF) accelerator facility consists of five cyclotrons, including a superconducting ring cyclotron, and two linear accelerators [1]. In the RIBF, we constructed a distributed control system based on the Experimental Physics and Industrial Control System (EPICS) for electromagnet power supplies, beam diagnostic instruments, vacuum control systems, etc. [2]. In FY2016, we started a new project at RIKEN RIBF to further advance synthesis of super-heavy elements with atomic numbers greater than 119, and the main points are as follows [3]:

A superconducting linear accelerator (SRILAC) is newly installed in the downstream part of the RIKEN linear accelerator (RILAC) to enhance the beam energy [4]. To increase the beam intensity for RILAC, the existing 18-GHz electron cyclotron resonance ion source (ECRIS) [5] is also upgraded to a new superconducting ECRIS (SC-ECRIS) [6].

Of these two, the new SC-ECRIS has the same structure as the RIKEN 28-GHz SC-ECRIS installed in the upstream of RILAC2 [7], which is one of the other injectors of the RIBF currently being operated. Therefore, as the devices to be controlled are almost the same as the RIKEN 28-GHz SC-ECRIS, the new SC-ECRIS control system should be constructed based on the current RIKEN 28-GHz SC-ECRIS control system with several improvements to overcome the limitations of the present system. In this proceeding, we discuss disadvantages of the current RIKEN 28-GHz ECRIS control system and report the design of the newly constructed control system in detail.

† a-uchi@riken.jp

RIKEN 28-GHZ SC-ECRIS CONTROL SYSTEM

System Concept

As the main feature of the RIKEN 28-GHz SC-ECRIS control system, Programmable Logic Controllers (PLCs) of the Yokogawa FA-M3 series and EPICS are utilized. The detailed system chart is shown in Fig. 1. The control system is mainly divided into two parts. One is F3RP61-2L, which is a CPU running Linux, to provide the operation services such as controlling of gas valves and power supplies [8]. In the case of the F3RP61-2L-based PLC station, EPICS is installed on the Linux-running system as the PLC CPU's operating system and the EPICS Input/Output Controller (IOC) is implemented as the middle layer for operation services, the so-called embedded EPICS. The other is the part of the implementation of the interlock function as a safety system [9]. Because the sequence CPU-based PLC station has the real-time feature, it is suitable for constructing the safety system, in which not so fast response, such as less than 1 ms, is required. Therefore, the interlock function is realized by the sequence CPU-based PLC station independently from the Linux-CPU-based PLC station, and the state of interlock is monitored by another external EPICS IOC via the TCP/IP network.

As the interlock, the system realizes the function of safely turning off the high-voltage power supply at the time of door opening, turning off the radio frequency (RF) at the time of vacuum abnormality.

Communication between PLC Stations

In the RIKEN 28-GHz SC-ECRIS control system, the main station, which is the installed Linux PLC CPU, manages four substations connected by fieldbus communication electrically isolated by optical fibers. In the case of the Yokogawa FA-M3 series, the fieldbus is called the FA bus. They communicate through FA-bus modules. Because heavy ions generated by an ion source are extracted to the low-energy beam transport by high voltage,

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP30

substations also need to be implemented at the high-voltage stage in some cases. As a typical example, the power supplies for the high-temperature oven method [10] and BIAS disk method [11] need to install the PLC substation on the high-voltage stage. Therefore, the optical fiber, being an insulator, is adopted for fieldbus communication between the main station and the substations. Consequently, the signal exchange between the sequence PLC CPU-based station for interlock and the Linux CPU-based PLC station needs to use EPICS Channel Access (CA) via the TCP/IP network or electrical signal.

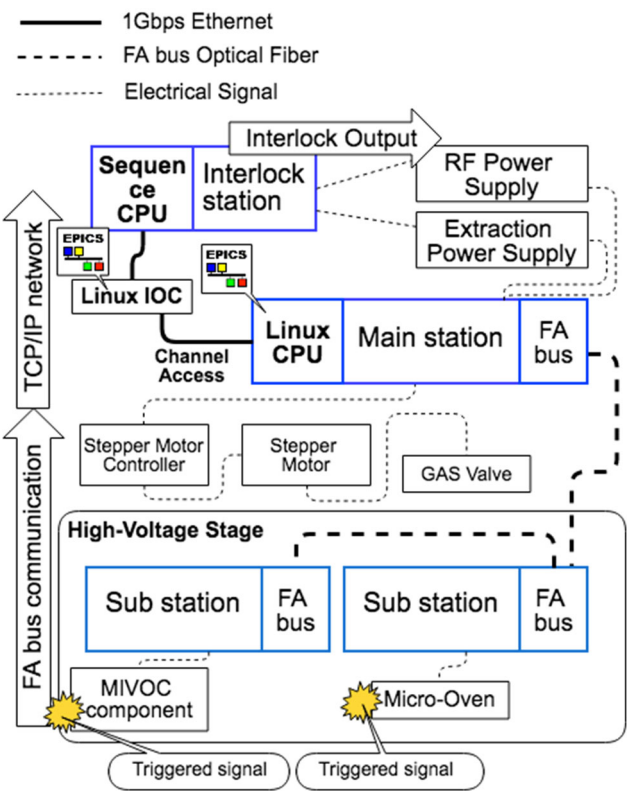


Figure 1: System chart of RIKEN 28-GHz SC-ECRIS control system. CA protocol is utilized for operation services and triggered signal.

Reliability of TCP/IP-based Interlock

There are two methods for sending the interlock signal from the Linux CPU PLC station to the sequence CPU PLC station, viz., sending an electrical signal to the sequence CPU PLC station via the Linux PLC's output module and sending the interlock signal by EPICS CA via TCP/IP. When using state information of the substation implemented in the high-voltage stage as an interlock signal, this system has the means to exchange signals only by the EPICS CA via the TCP/IP network. In general, the reliability of the CA-based interlock system is not high because of the failure of the network switch in the network route, the problem of slow signal transmission speed compared with the bus access, and the problem of reliability of the EPICS IOC. Thereby, the reliability of the signal through TCP/IP is lower than that of the electric signal, and thus the reliability of the interlock is also not relatively high.

NEW SC-ECRIS CONTROL SYSTEM

System Design

In the new project, a new SC-ECRIS is installed upstream of RILAC to increase the beam intensity for RILAC. The photograph of the new SC-ECRIS for SRILAC is shown in Fig. 2. Considering the new SC-ECRIS for SRILAC, the control system should follow the current RIKEN 28-GHz SC-ECRIS control system upstream of RILAC2, because the RIKEN 28-GHz SC-ECRIS control system has achieved success in the RIBF project. Thus, in the case of the new SC-ECRIS control system, we have adopted the Yokogawa FA-M3V series, which is the upgraded FA-M3, for system construction. On the other hand, we should also solve the disadvantage of low reliability for interlock features in the RIKEN 28-GHz SC-ECRIS control system when constructing the new SC-ECRIS control system. Accordingly, to solve the disadvantage, a new SC-ECRIS control system has been designed by implementing two different types of CPUs in the main PLC station. Essentially, the sequence PLC CPU in the first slot and the Linux PLC CPU in the second slot have been implemented in the same PLC base module. In the sequence PLC CPU, the ladder program runs for the interlock system, and the Linux CPU runs the EPICS IOC and provides operation services to users via the EPICS CA protocol. Currently, the new SC-ECRIS control system consists of a main PLC station and five PLC sub stations with star-topology fieldbus communication using optical FA bus modules. The detailed system diagram is shown in Fig. 3.

At present, this control system does not include the control of superconducting electromagnet power supplies. Control of superconducting electromagnet power supplies is implemented in a system consisting of another controller and client without EPICS IOC. The system implementation test with EPICS is in progress now.

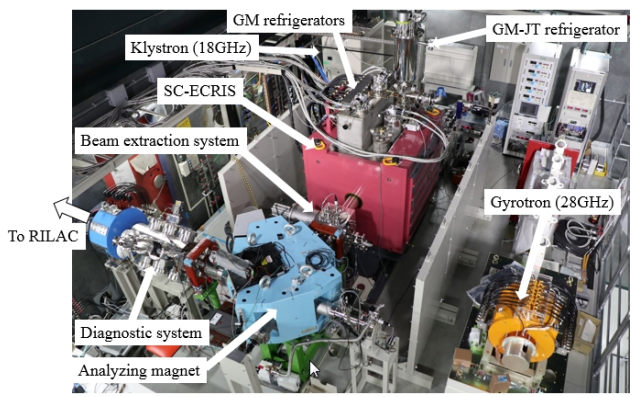


Figure 2: Newly installed SC-ECRIS for SRILAC. Currently, an 18-GHz RF source (Klystron) is only used for the beam commissioning [5].

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP30

Interlock

Generally, X-rays are generated during the operation of the ion source and the high voltage is utilized for extraction of heavy ions generated. Therefore, this interlock has to prevent workers from exposure to X-ray and/or accidental electrification.

Interlocks are roughly divided into two types. One is a human protection system, which has a mechanism to turn off the RF power and power supplies for high-voltage beam extraction when a person enters the ion source room. This interlock system utilizes the opening of doors and entrance information as the triggers. The other interlock is a machine protection system. The system monitors the cooling water, vacuum condition, etc., and automatically stops the ion source and the devices constituting it, safely in case of abnormality.

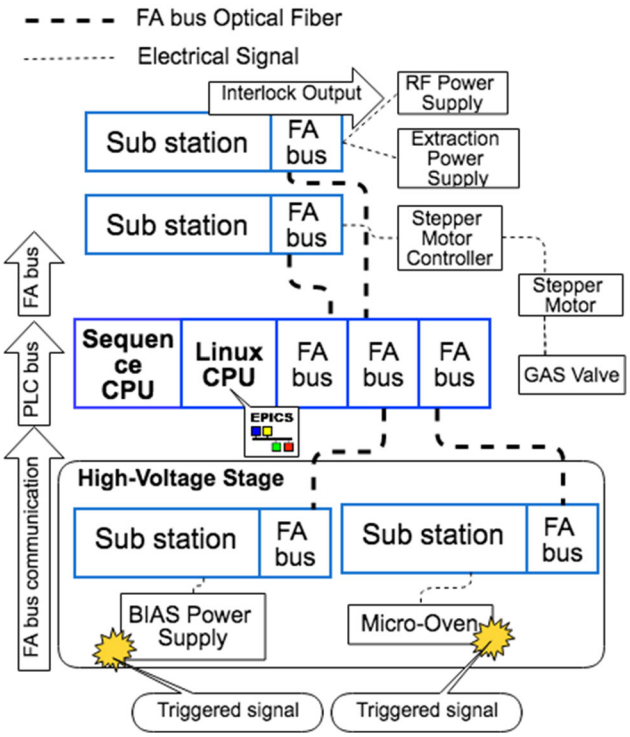


Figure 3: System chart of new SC-ECRIS control system for SRILAC. Interlock signal is delivered by bus communications.

I/O Sharing Method

The feature of the Yokogawa FA-M3V series is that, as a behavior when there are two CPUs, reading can be done from both CPUs unconditionally, but direct instructions to the output module adopt exclusive control. Therefore, the basic write (output control) is performed only from the sequence PLC CPU, when performing output control from the EPICS IOC on the Linux PLC CPU; it will be realized via internal registers of the sequence PLC CPU, for example, I00001.

I/O Refresh Time for FA Bus using 0ptical Fiber

According to Reference [12], the I/O refresh time for each access type (read/write) via the optical FA bus module is estimated to be the product of the access time and "Number of modules converted to a 16-point basis". The access time is the sum of "Time dependent on access type" and "Time dependent on transmission distance". Based on this estimation, the theoretical transfer I/O refresh time is 16.8 μs in this system. This value is sufficiently higher than the TCP/IP-based network speed.

System Availability

Based on the status information of the doors and entrance, we are able to provide a human safety mechanism as an interlock system of RF source and extraction voltage of SC-ECRIS. Accordingly, it is possible to realize the interlock signal for the machine protection system via FA bus communication without going through the TCP/IP network, even though the control of power supply for the oven system is mounted on the high-voltage stage. As the system behavior, the oven power supply is turned off by triggered by the degree of vacuum and cooling water temperature.

CONCLUSION

We solved the disadvantage of the insufficient reliability of the interlocks in the RIKEN 28-GHz SC-ECRIS control system and constructed a control system of the new SC-ECRIS for SRILAC. This new SC-ECRIS control system was successfully used in the test operation of the new SC-ECRIS performed in August 2018 without any serious problem. Because of the mounting of two CPUs in one base unit, it is possible to exchange the trigger signal for interlocking with the sequence PLC CPU from the Linux PLC CPU via the FA bus on the PLC base module. Therefore, it is possible to share the interlock signal of the high-voltage stage with the Linux PLC CPU and the sequence PLC CPU without going through the TCP/IP network, and it improves the system reliability of the interlock feature successfully without lowering the conventional system usability.

REFERENCES

[1] O. Kamigaito *et al.,* "Present Status and Future Plan of RIKEN RI Beam Factory", in *Proc. IPAC'16*, Busan, Korea, May 2016, pp. 1281-1283.

[2] M. Komiyama *et al.,* "Status of the RIKEN RI Beam Factory Control System", in *Proc. ICALEPCS'13,* San Francisco, CA, USA, Oct. 2013, pp. 348-351.

[3] H. Okuno *et al.,* "Operational Experiment and Upgrade Plans of the RIBF Accelerator Complex", in *Proc. Cyclotrons'16,* Zurich, Switzerland, Sep. 2016, pp. 1-6.

[4] N. Sakamoto *et al.*, "Construction Status of the Superconducting Linac at RIKEN RIBF", in *Proc. LINAC'18*, Beijing, China, Sep. 2018, paper WE2A03.

[5] K. Ozeki *et al.*, "SUPPLY OF METALLIC BEAMS FROM RIKEN 18-GHz ECRIS USING LOW-TEMPERATURE OVEN", in *Proc. HIAT2015,* Yokohama, Japan, Sep. 2015, pp. 244-246.

[6] T. Nagatomo *et al.*, "New 28-GHz Superconducting ECR Ion Source for Synthesizing New Super Heavy Elements of

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP30

Z > 118", presented at ECRIS'18, Catania, Italy, Sep. 2018, paper TUA3, unpublished.

[7] Y. Higurashi *et al.*, "Recent Development of RIKEN 28 GHz SC-ECRIS", in *Proc. ECRIS'16*, Busan, Korea, Aug.-Sep. 2016. pp. 10-13.

[8] M. Komiyama *et al.*, "Upgrading the Control System of RIKEN RI Beam Factory for New Injector", in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, pp. 275-277.

[9] A. Uchiyama *et al.*, "Construction of Client System for 28GHz SC-ECRIS", RIKEN Accelerator Progress Report vol.43 (2010), p. 133-134.

[10] J. Ohnishi *et al.*, "Development of a high-temperature oven for the 28 GHz electron cyclotron resonance ion source", *Review of Scientific Instruments*, 85, 02A941 (2014).

[11] Y. Higurashi *et al.*, "Production of a highly charged uranium ion beam with RIKEN superconducting electron cyclotron resonance ion source", *Review of Scientific Instrument*s, 83, 02A333 (2012).

[12] Fiber-optic FA-bus Module User's Manual;
https://web-material3.yokogawa.com/IM34M06H45-01E.us.pdf