ISBN: 978-3-95450-231-8 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2023-THPA121

A PUBLIC DATA SERVICE FOR THE BEAM INTERLOCK SYSTEMS AT CERN - CURRENT STATUS AND FUTURE PLANS

J. C. Garnier, J. F. Barth, A. Buszydlik, M.A. Galilee, T. Skarhed, D. Wollmann
CERN, Geneva, Switzerland

Abstract

The Beam Interlock System (BIS) is the backbone of the machine protection system in CERN's accelerator chain, ensuring that the beams are safely transported through the injector chain and circulated in the Large Hadron Collider (LHC). A new version of the BIS is currently under development and planned to be deployed in the Super Proton Synchrotron (SPS), LHC and the North Area experimental zone during Long Shutdown 3 (LS3), while the recently installed BIS in LINAC4 and the PSB will remain in place. As a result, the current and the new system will be operated in parallel, and it is fundamental that both systems can be supervised and monitored in the same way by the operation crews, the system experts, and reliability engineers. Consequently, it is planned to provide a data service with a unique Application Programming Interface (API) for both systems. This data service will leverage Unified Controls Acquisition and Processing (UCAP) and chain transformations to expose data for anyone to consume, and to be logged as time series in the CERN Accelerator Logging Service (NXCALS). This paper recalls the current implementation of the BIS supervision. It then presents the solution that was developed with UCAP and the benefits of the chain of transformations, and draws an outlook of the future plans.

INTRODUCTION

The particle beams and the magnet circuits of the LHC [1] store unprecedented amounts of energy. An uncontrolled release of this energy would cause significant damage to the accelerator, requiring extensive repairs which considerably reduce the available time of the accelerator to produce physics [2] and can also be costly.

The BIS [3] constitutes a crucial part of the machine protection systems for the LHC and its injector chain. Accelerator subsystems [1] like the radio frequency, vacuum, beam instrumentation, collimation system etc., are connected to the BIS and allow operation only if certain conditions are met. If these conditions are not met, the beams are extracted via the beam dumping system [4, 5] or other actuators to protect the accelerator against damage. Values from these subsystems are called user permits. They are combined to calculate the value of the so-called Local Permits and Beam Permits, which are then used to evaluate the accelerator-wide Global Beam Permit. Certain user permits may be masked (overridden) by the operator under certain conditions, or completely disabled at the hardware level. This influences the results of the Local and Global Beam Permits. The BIS hardware boards expose their data to the control software with a Versa Module Eurocard (VME) bus [6]. It is important to note that the BIS performs all its safety critical tasks independently of any control software layer. The BIS control system is then responsible to expose the data to the users of the BIS, and to forward commands to the hardware boards.

The new version of BIS [7, 8] is currently in the prototype phase and will be deployed in the SPS, LHC and the North Area experimental zone during Long Shutdown 3 (LS3), starting 2026. The recently installed BIS in LINAC4 (Linear Accelerator 4) and the PSB (Proton Synchrotron Booster) will remain in place. The Controls and Beam Studies for Protection section of the Machine Protection and Electrical Integrity Group[1] is developing the control software for BIS v2. Both versions of the BIS will be used in parallel. The aim is to provide a common interface for both BIS systems to ease the integration and the maintenance of the system in operation.

The first section of this paper explains the current use cases of the system based on the experience of exploiting the first version for two runs, and on discussions with operators. The second section outlines the modernized architecture. Finally, the vision for the future is discussed.

USE CASES

Users of the Beam Interlock System can be categorized as follows:

- BIS experts, designers of the systems and responsible for its correct behaviour during operations and its maintenance;
- Operators, who run the accelerators and their numerous subsystems from the controls centre;
- Availability and reliability experts, for which the BIS is particularly interesting as it aggregates permits from the subsystems of the accelerators;
- Experts of systems providing inputs to the BIS, to verify that their system is behaving properly during commissioning.

Experience from LINAC4, PSB, SPS and LHC Operation

The BIS control system typically relies on a thin real-time layer that performs the actual read/write access to the hardware via the VME bus, and exposes data in the BIS hardware encoded format. This is done in order to keep this layer very simple, and to limit the needs to change it during

[1] https://mpe-cb.web.cern.ch

ISBN: 978-3-95450-231-8 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2023-THPA121

the lifetime of the system. Consequently, the decoding of the data is performed in higher layers.

In the original BIS control software architecture, applications that wanted to work with data from the BIS devices needed to rely on the decoding library provided in Java. This strategy worked when the control system was composed exclusively of Java. A limitation is that with every change of the device API, the decoding library needs to be updated and the client software to be recompiled and redeployed. Another limitation is that the BIS produces numerous values that are of interest for trend analysis, and this strategy did not cover the use case of logging data into NXCALS [9].

Reactive streams [10, 11] were introduced to provide a flexible, scalable and reusable way to manage the data. It implemented the pattern Acquire-Tranform-Publish. Users could then consume decoded data from virtual devices using a converter of streams to virtual devices. This allowed the data to be logged in NXCALS in a decoded way, and to be used conveniently for data analysis and presentation by any user. With changes on the devices however, the streams would have to be adapted and the virtual devices redeployed. Clients subscribing to properties would not need to change if the API remained compatible. The maintenance was primarily on the BIS software developer side. This solution was implemented in 2015, before UCAP [12] was designed.

Public Usage of Data

The data from the BIS is of primary importance during the commissioning and the operation of the accelerators.

During the commissioning, various systems notably assert that they send an interlock to the BIS under certain conditions. To perform this assertion, they either rely on offline data stored by the BIS in NXCALS and Post Mortem [13], or on online data coming from the BIS control system. Numerous commissioning solutions rely on online data and can leverage a public data service. This way they can be implemented in any language.

During operations, publicly available decoded data is used for fixed displays in the control rooms. Users also wish to use this data to complement the Graphical User Interface (GUI) of their systems with information coming from the BIS.

Furthermore, publicly available decoded data would allow automatizing certain tasks and procedures.

ARCHITECTURE

The aim is to expose a maintainable and documented API based on the standard CERN device-property model [14]. Device and property names must be meaningful and easily identified by potential users.

UCAP will be used for the implementation of the future control system of the BIS and BIS v2. It brings advantages compared to the current streams:

- Widely accepted solution for the Acquire-Transform-Publish pattern at CERN;
- Implementation of virtual devices following the standard device-property model;
- Internal support provided;
- Off-the-shelf supervision and monitoring of the infrastructure;
- Possible extensions by the users.

The BIS and BIS v2 hardware typically exposes data in two ways:

- Registers, which are a memory map of 32 bit values;
- History buffers, which are ring buffers that contain the latest events recorded by the hardware.

The public API decodes data coming from either the registers or the history buffers, or combines them, and exposes them in a user-friendly way.

UCAP makes use of *converters*: pieces of code that can be configured and describe the data transformation, as presented in Fig. 1. Converters are supposed to be generic, in order to provide greater re-usability. The developed converters have been made as generic as possible, to allow for their use by other systems than the BIS.

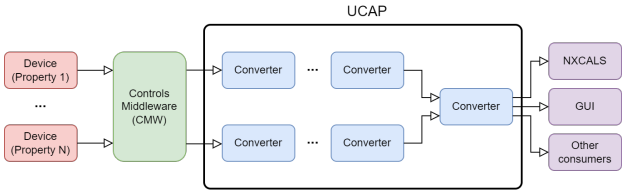

Figure 1: UCAP converters acquire data from CMW devices or other converters, transform it, and republish it.

The HistoryBufferExtractor converter's responsibilities are to filter record event types according to their given configuration, then to combine the time fields to construct a single nanosecond timestamp, and then to republish the buffer updates as separate values.

Based on the experience of the streams solution, the UCAP converter makes use of specification JSON [15] files to specify how to decode the registers. One can map bits of the registers to a named and typed data field, that will be exposed by the UCAP virtual device. This was done in a way that makes it easy for systems maintainers to create and update decoders.

The RegisterDecoder converter's responsibilities are to read a register from a hardware device, decode its content according to the specifications, and publish the result as a single value.

In some cases, the history buffer may fail to produce an updated value. This can happen, if the event went out of the ring buffer before it was read by the supervision software, or if the electronics board was just initialized and the history buffer is basically empty. When this happens, a fallback

ISBN: 978-3-95450-231-8 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2023-THPA121

value may be used, coming from the registers, as described in Fig. 2.

The LatestValueCombinator converter's responsibilities are to publish values from the main subscription, or to use a fallback subscription in case the main subscription fails to provide a value, for any reason.

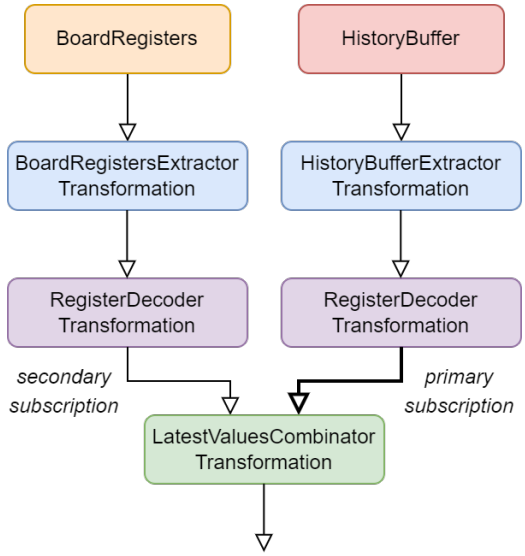

Figure 2: `LatestValuesCombinatorTransformation` allows selecting between two independent data sources.

UCAP provides a Java API for generating virtual device configuration files, on which an abstraction has been built upon. This heavily reduced code duplication, while simplifying configuration of multiple virtual devices that are chained together into pipelines. Configuration as code has been implemented. It provides pre-configured transformations for the BIS use case, that could be extended to any other use cases. It describes the processing pipeline that take the source device as input and chain all the transformations that must be applied to publish the final property. Last but not least, it generates different configurations for our different environments: production, staging, testbed.

At the moment, multiple options are considered for defining virtual UCAP devices.

Firstly, the virtual devices could mirror the existing hardware devices one to one. Consequently, the class of such virtual device would expose numerous properties and fields. This would lead to more complicated maintenance, new features would need to be added as new properties in this class, therefore impacting the existing and running devices.

Secondly, a virtual device can provide one and only one feature from the hardware device. The class of such a device would remain simple, with typically one property representing the specific feature of the hardware device. Adding new features, or refactoring existing features, would not impact any other device-properties, which is a great advantage and gives a lot of flexibility for the maintenance and evolution of the system.

A third case is that certain virtual devices will aggregate information from multiple BIS hardware devices and propose a higher level of abstraction, enabling users to interact only with this higher level of abstraction. The entire BIS installation of an accelerator could be controlled by such high level virtual device, to dispatch multiple commands to unitary devices or to aggregate overall statuses of the installation.

OUTLOOK

The first UCAP converters have been implemented and are used operationally. They run in parallel to the legacy stream-based service, in order to assert the correctness of the logging data in comparison to data logged from the legacy service. The aim is to eventually provide the data that will be used by all types of users, the GUIs for operators and experts, data analysis, etc. At this stage, key information is already provided for the current BIS: beam permits and local permits, user permits, masks and disable statuses. The UCAP architecture looks very promising in terms of maintainability and extensibility. Additional information still needs to be exposed in order for the BIS control system to be entirely ported to UCAP. Seeing the potential of the solution, more extensions are being required by the users to provide more automations to the operations and analysis.

The work performed up to now is solely based on the control system of the current BIS. With the coming of BIS v2, an additional layer of transformation in UCAP will be provided so that operations and data analysts can interact with both systems.

UCAP also opens the possibility to any users to implement their own converters. While these contributions are very welcome, it will be a challenge to make them reusable for any BIS installations, as they will need to be identified and streamlined.

CONCLUSION

Based on the experience with the Beam Interlock System over several years and the inputs and feedback from operation crews, together with the availability of new systems, a new control system is currently being designed for the current BIS and the upcoming BIS v2. The data acquisition pipeline of the control system was designed to be extensible with minimum effort, using configurable generators helping the user to define rules to decode BIS registers and history buffers. Using this software stack, an abstraction layer is proposed to the users so that they can work transparently with BIS and BIS v2.

REFERENCES

[1] L. Evans and P. Bryant, "The CERN Large Hadron Collider: Accelerator and Experiments - LHC Machine", *J. Instrum.*, vol. 3, p. S08001, 2008. doi:10.1088/1748-0221/3/08/S08001

ISBN: 978-3-95450-231-8 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2023-THPA121

[2] R. Schmidt *et al.*, "LHC Machine Protection", in *Proc. PAC'07*, Albuquerque, NM, USA, Jun. 2007, paper TUZAC03, pp. 878–882.

[3] B. Puccio, A. Castañeda, M. Kwiatkowski, I. Romera, and B. Todd, "The CERN Beam Interlock System: Principle and Operational Experience", in *Proc. IPAC'10*, Kyoto, Japan, May 2010, paper WEPEB073, pp. 2866–2868.

[4] P. Van Trappen *et al.*, "SPS Beam Dump System (SBDS) Commissioning After Relocation and Upgrade", in *Proc. IPAC'22*, Bangkok, Thailand, Jun. 2022, pp. 2530–2532. doi:10.18429/JACoW-IPAC2022-THPOST039

[5] E. Carlier *et al.*, "LHC Beam Dumping System", in *Proc. 7th Evian Workshop on LHC Beam Operation*, Evian, France, 2017, pp. 215-220.

[6] J. A. Black, *The System engineer's handbook: a guide to building VMEbus and VXIbus systems*, Morgan Kaufmann.

[7] R. L. Johnson, C. Martin, T. Podzorny, I. Romera, R. Secondo, and J. A. Uythoven, "The Consolidation of the CERN Beam Interlock System", in *Proc. IPAC'21*, Campinas, Brazil, May 2021, pp. 3309–3312. doi:10.18429/JACoW-IPAC2021-WEPAB282

[8] I. Romera *et al.*, "Design considerations for CERN´s second-generation Beam Interlock System", presented at the IPAC'23, Venice, Italy, May 2023, paper THPA062, this conference.

[9] J. P. Wozniak and C. Roderick, "NXCALS - Architecture and Challenges of the Next CERN Accelerator Logging Service", in *Proc. ICALEPCS'19*, New York, NY, USA, Oct. 2019, pp. 1465–1469. doi:doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

[10] M. A. Galilée *et al.*, "Renovation and Extension of Supervision Software Leveraging Reactive Streams", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 1753–1756. doi:10.18429/JACoW-ICALEPCS2017-THPHA152

[11] A. Calia *et al.*, "Streaming Pool - Managing Long-Living Reactive Streams for Java", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 1837–1841. doi:10.18429/JACoW-ICALEPCS2017-THPHA176

[12] L. Cseppentő and M. Büttner, "UCAP: A Framework for Accelerator Controls Data Processing @ CERN", in *Proc. ICALEPCS'21*, Shanghai, China, Oct. 2021, pp. 230–235. doi:10.18429/JACoW-ICALEPCS2021-MOPV039

[13] J. F. Barth *et al.*, "A Modernized Architecture for the Post Mortem System at CERN", in *Proc. IPAC'22*, Bangkok, Thailand, Jun. 2022, pp. 1557–1560. doi:10.18429/JACoW-IPAC2022-TUPOMS055

[14] A. Dworak *et al.*, "The new CERN Controls Middleware", *J. Phys.: Conf. Ser.*, vol. 396, p. 012017, 2012. doi:10.1088/1742-6596/396/1/012017

[15] JSON, http://www.json.org/