ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-THPG30

IMPROVEMENT OF THE LHC ORBIT FEEDBACK TESTING FRAMEWORK

A. Calia, D. Alves, J. Wenninger, M. Hostettler, S. Jackson
CERN, Geneva, Switzerland

Abstract

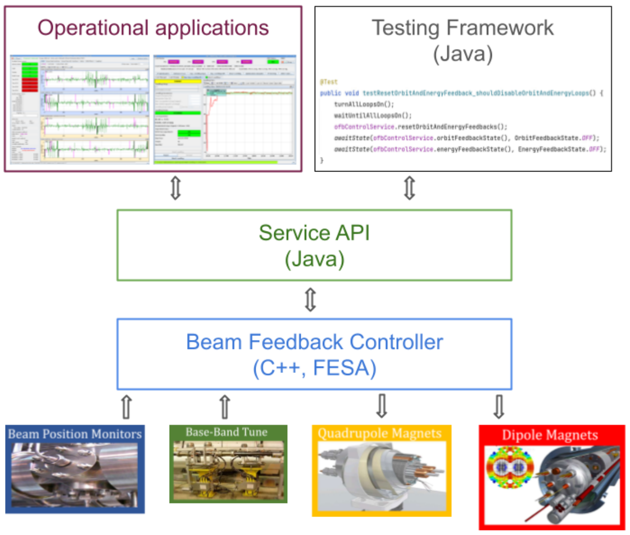
During the Long Shutdown 2 (LS2 2019-2021), the orbit feedback correction software (OFB) of the LHC was redesigned to satisfy new requirements for Run 3 (2022-2025) and to clean up legacy functionalities. The OFB is an essential component of LHC high intensity operation since the orbit must be stabilized to a fraction of the beam size during the entire LHC machine cycle and reproducibility is key for efficient operation. Redesigning such an essential and complex system during shutdowns requires thorough testing of the system functionality. The existing OFB testing system has been reviewed and improved based on the experience of LHC Run 2. An automatic, continuous integration tool has been put in place to validate future software developments before putting them in production. The solution for the OFB testing will be presented in this contribution.

INTRODUCTION

The Large Hadron Collider (LHC) is a particle accelerator and collider that is currently operating with two counter-rotating beams of protons or ions at a record beam energy of 6.8 TeV. Given the high stored beam energy, machine tolerances are very tight to avoid potential damage due to uncontrolled particle loss. An automatic feedback system is therefore essential for the operation of the LHC.

The full-stack system (from data acquisition to integration with the LHC controls system) responsible for simultaneously controlling the beam orbit and the machine tune is the so-called Orbit Feedback system (OFB). The beam orbit is kept under control by minimizing the difference between the measured orbit and a reference. The machine tune is a crucial parameter to control particle loss and is also kept on the reference value with the OFB. Beam orbit perturbances occur due to multiple factors, such as magnet misalignments, ground motion, earth tides, and machine configuration changes (optics). Tune transients occur during dynamic phases in an LHC fill, such as the energy ramp and optics transitions. Due to uncontrollable magnetic field errors, the transition from the injection energy, 0.45 TeV, to 1 TeV of the ramp is particularly violent for the tune of the beams. During this phase, the tune feedback system is essential for keeping the tune of the beam on the reference value.

Controlling the orbit and tune of the LHC requires a real-time software architecture. The implementation is realized in C++ using the Front-End Software Architecture (FESA) framework [1]. The FESA framework is a software suite developed at CERN to design and implement real-time software and deploy it on front-end computers. The real-time component of the OFB, a FESA class, is called the Beam Feedback Controller (BFC). This FESA class receives beam position data from over 500 Beam Position Monitor (BPM) sensors per beam (measuring horizontal and vertical positions) and 6 tune measurement devices per beam [2] With a control loop frequency of 25 Hz, the BFC calculates the required real-time corrections to orbit and tune. Around 500 dipole corrector magnets per beam correct the orbit, RF frequency adjustments compensate for radial movements, and over 300 quadrupole corrector magnets correct the tune [3]. An overview of the various layers of the OFB is shown in Fig. 1.



```
@Test
public void testResetOrbitAndEnergyFeedback_shouldDisableOrbitAndEnergyLoops() {
    turnAllLoopsOn();
    waitUntilAllLoopsOn();
    ofbControlService.resetOrbitAndEnergyFeedbacks();
    awaitState(ofbControlService.orbitFeedbackState(), OrbitFeedbackState.OFF);
    awaitState(ofbControlService.energyFeedbackState(), EnergyFeedbackState.OFF);
}
```

Figure 1: Schematics of the OFB software stack. At the bottom, the LHC systems: BPM, Base-Band Tune, quadrupole and dipole magnets. The BFC FESA class interacts with these systems to control the orbit and tune. The Java Service API orchestrates the interaction of operational applications and the testing framework with the FESA class.

The entire OFB system includes a Java service layer, a testing framework, and the necessary adapters needed to integrate the system into the LHC controls system. This contribution discusses the renovation of each layer performed during LS2 (after LHC Runs 1 and 2), with particular emphasis on the testing framework.

RENOVATION OF THE BFC SYSTEM DURING LS2

During LS2, the decision was taken to consolidate the FESA code initially split into two separate processes [4]. This was made possible by a hardware upgrade from a 24-core, 32 GB RAM, 15 MB cache machine to a 64-core, 200 GB RAM, 22MB cache machine, allowing more advanced code to be implemented while retaining a deterministic execution. Furthermore, the FESA framework improved significantly, allowing the use of many new functionalities and simplifying the FESA class codebase.

As a result of this major effort, the maintainability of the BFC FESA class code also improved significantly, and because it is based on standard FESA features, porting to future versions of the FESA framework will be easier.

Alongside code cleanup and improvements, two new features were requested by the LHC operations team based on the experience of LHC Run 1 and 2. The most notable feature requests were:

- Introduce a function player to be able to change the BFC parameters following LHC dynamic phases, instead of relying on static values
- Review the optics calculation procedures to speed up the preparation of the LHC cycle and improve the overall stability of the system

During the LHC's dynamic phases, for example the energy ramp from 0.45 TeV to 6.8 TeV, the machine configuration (optics and reference orbit) changes significantly. The new function player allows the LHC operations team to specify functions for the reference orbit, reference tune, and multiple gain factors. With this capability, the BFC real-time corrections can more closely follow the actual LHC machine state and significantly improves the overall performance of the feedback loops.

The optics of the machine are also an input of the orbit feedback loop, as they are used to calculate the Pseudo-Inverse of the Response Matrix (RM) needed to derive the orbit corrections to be sent to the dipole corrector magnets. In the new version of the BFC FESA class, the parameters of the optics (Twiss parameters) are uploaded during every preparation for beam for the entire cycle of the LHC. Internally, the BFC calculates the corresponding matrices and stores them in memory. The function player will then be instructed to switch from one optic to the next to follow the LHC cycle's dynamic changes. Given the nature of the calculations of the matrices, a Hardware Acceleration feasibility study was performed to increase the performance. It was concluded that GPUs did not improve computation times when calculating the Pseudo-Inverse (Pinv) of the RMs compared to a multi-core approach [5].

NEW SERVICE SOFTWARE ARCHITECTURE FOR THE OFB

In the LHC ecosystem, Java is the programming language that offers seamless integration with the various aspects of the controls system, like the settings management system (LSA), the sequencer (used daily to control the LHC cycle) and the CERN device communication protocol (JAPC/RDA3).

During LHC Run 1, it was decided to create a Java abstraction layer over the BFC FESA class to offer better integration

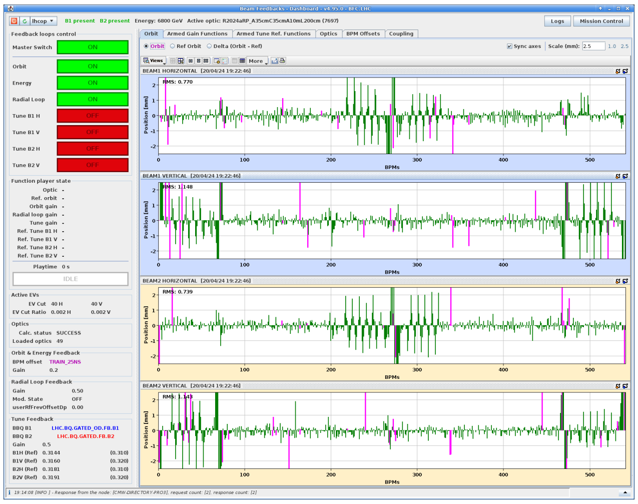


Figure 2: OFB Dashboard application used daily in LHC operation with BPM readings.

with the LHC controls system, the OFB Java service API. During LS2, given the experience of Run 1 and Run2 [6], this service layer was completely renovated with a modern architecture leveraging the new BFC functionalities.

The Java API is divided into two levels:

- Delegates: Java classes responsible for communication with the BFC FESA class. Each delegate exposes the API of the BFC via Java domain objects.
- Services (or API): Java classes combining one or more delegates to achieve a high-level task with a simple API. Via services, users can perform complex actions that require multiple interactions with the BFC FESA class.

The OFB Java service API is now integrated with various LHC applications. The main advantage of this solution is that there is a single API to interact with the BFC FESA class, enhancing its functionalities and easing migration to future BFC versions.

An example of an application built with this API is the OFB Dashboard, used in the LHC CERN Control Centre (CCC) to control various aspects of the feedback systems. A screenshot of the application is shown in Fig. 2.

IMPROVEMENT OF THE TESTING FRAMEWORK

The OFB system plays a crucial role in the performance and availability of the LHC. Therefore, proper testing prior to deployment to production is mandatory to ensure the correct functioning of the system.

A testing framework, written in Java, was developed during Run 1 of the LHC [7]. The ideas behind this framework were used as the basis for a new testing framework design. This new framework highlighted limitations of the previous implementation of the BFC FESA class. In the past, only basic testing could be done and the emphasis on being able to express tests in an eDSL (embedded Domain-Specific

Language) increased the difficulty of evolving the testing framework.

During the refactoring of the BFC FESA class in LS2, the testing framework was entirely rewritten, and testing capabilities were taken into account at every step of the FESA class development. The new version of the testing framework uses the Java service API to ensure that tests cover the full stack: the BFC FESA class behavior, its API, and the Java service layer. The testing framework is based on the Java JUnit framework to start and record test executions.

In order to enable testing, the new version of the BFC FESA class implements a special feature to enable simulation mode. In this mode, the software is able to run standalone without the need for LHC-specific timing events. It is therefore possible to simulate the 40ms tick of the control loop and the LHC energy read back.

It is no surprise that writing tests at the same time as expanding the features of a system is beneficial for the final result [8]. Test Driven Development (TDD) is a common practice in software development. Currently, the test suite of the LHC OFB system includes more then 100 tests. These are system tests following a black-box principle, treating the BFC FESA class as an independent entity and purely using its public API for access. Tests can be run on-demand, and a report with the results can be generated. This feature is useful when validating the behavior of a specific version of the BFC FESA class, before deployment to production. Tests are also run in a Continuous Integration (CI) fashion. In this case, the capabilities of the Gitlab CI pipeline (the product used at CERN for Git projects) are used. In this configuration, the test suite runs for every new commit on the Java service API or the BFC FESA class. Execution of code outside of the CERN production environment is ensured by the use of standard Open Container Initiative (OCI) containers [9]. Containers are a way to package software along with every dependence, including the operating system.

In this context, a container for Java Service API (plus testing code) and another with the BFC FESA class are created. Containers in Gitlab CI pipelines are first class citizens. Every commit on the Java Service API project or the BFC FESA class creates a new container with the latest version of the code. Once the containers are ready, an ad-hoc pipeline is triggered to run the tests. Using containers for FESA classes is not a common practice due to the difficulty to "containerize" C++ code with direct access to Hardware (for example timing cards or FPGAs). The BFC container is, therefore, limited in its capabilities (it is currently not conceivable to run the container in production).

However, the testing framework aims to validate API compliance and the behavior of the main loop of the BFC FESA class, not its real-time performance. Given the scope, the container approach proved to be a flexible enough solution to run tests targeting a FESA class outside the production environment while still guaranteeing an adequate level of testing capabilities and confidence in the results.

LESSONS LEARNED AND FUTURE OUTLOOK

The development of the testing framework proved to be challenging. The infrastructure around it (API and Gitlab configuration) must be carefully configured. The BFC FESA class needed to be adapted to have a simulation mode, and its containerization needed to be explored and prototyped.

The investment is considered to be worthwhile, given the quality of the refactoring. Black-box testing during development allowed for a clean and understandable API that could be integrated without surprises into the LHC control system. The refactoring decisions could be steered in an iterative manner, resulting in no loss of development time.

The produced testing suite ensures that the LHC's OFB system is in operational condition before starting the LHC commissioning phase. Nevertheless, the configuration and behavior of the beam still need to be carefully tested. Simulating every aspect of the system is impossible; the tests can only be coded to detect known failure scenarios and regressions.

It is important to continue evolving the testing framework to cover scenarios as they are highlighted during LHC beam operation. A final, decisive benefit comes from the comprehensive test specification of the system. These specifications effectively document the behavior of the BFC FESA class and how it should function to work properly in production. They can be used as a compliance framework for every new refactoring of the BFC FESA class.

CONCLUSIONS

During Long Shutdown 2 of the LHC, the Orbit Feedback system underwent significant modifications, aimed at enhancing its performance, reliability, and maintainability.

A comprehensive testing framework was implemented to support code refactoring efforts, which also played a crucial role in expediting the development process. The testing framework systematically assessed the functionality of the revised codebase through automated tests, enabling developers to refine the system architecture and address potential vulnerabilities, thereby enhancing the overall robustness of the system.

The modifications implemented during LS2 resulted in a notable decrease in the commissioning time at the start of LHC Run 3. In particular, it significantly reduced the beam-time needed to commissioning the system since most of its functionalities were tested and validated without beam. Only a final check of the behavior with real-time data coming from instruments measuring the real beam was necessary. Configuration issues and real-time behavior needed some adjustments as these weaknesses are only highlighted with beam. Nevertheless, the OFB system functionally behaved as expected.

REFERENCES

[1] M. Arruat *et al.*, “Front-end Software Architecture”, in *Proc. ICALEPCS'07*, Oak Ridge, TN, USA, Oct. 2007, paper

WOPA04, pp. 310–312.

[2] M. Gasior and R. Jones, “High Sensitivity Tune Measurement by Direct Diode Detection”, in *Proc. DIPAC’05*, Lyon, France, June 2005, paper CTWA01, pp. 312–314.

[3] O. Brüning, P. Collier, P. Lebrun, S. Myers, R. Ostojic, J. Poole and P. Proudlock, “LHC Design Report”, CERN, Geneva, Switzerland, Rep. CERN-2004-003-V-1, 2004. doi:10.5170/CERN-2004-003-V-1

[4] R. J. Steinhagen, “LHC Beam Stability and Feedback Control - Orbit and Energy”, Ph.D. thesis, RWTH Aachen U., Rep. CERN-THESIS-2007-058, June 2007.

[5] L. Grech, D. Alves, S. Jackson, G. Valentino, and J. Wenninger, “Feasibility of hardware acceleration in the LHC orbit feedback controller”, in *Proc. ICALEPCS’19*, New York, NY, USA, Oct. 2019, pp. 584–588. doi:10.18429/JACoW-ICALEPCS2019-MOPHA151

[6] L. K. Jensen*et al.*, “Software Architecture for the LHC Beam-based Feedback System at CERN”, in *Proc. ICALEPCS’13*, San Francisco, CA, USA, Oct. 2013, paper THPPC119, pp. 1337–1340.

[7] S. Jackson, D. Alves, L. Di Giulio, K. Fuchsberger, B. Kolad, and J. Pedersen, “Testing framework for the LHC beam-based feedback system”, in *Proc. ICALEPCS’15*, Melbourne, Australia, Oct. 2015, pp. 140–144. doi:10.18429/JACoW-ICALEPCS2015-MOPGF024

[8] D. Alves, K. Fuchsberger, S. Jackson and J. Wenninger, “Test-driven software upgrade of the LHC beam-based feedback systems”, in *Proc. 20th IEEE-NPSS Real Time Conf.*, Padua, Italy, June 2016. doi:10.1109/RTC.2016.7543106

[9] Open Container Initiative, https://opencontainers.org