A monitoring system for the beam-based feedbacks in the LHC

Diogo Alves, Stephen Jackson and Iason-Dimitrios Rodis

***Abstract*—The system for the beam-based feedbacks in the LHC is one of the most complex in CERN's accelerator complex. It is an essential system for the operation of the LHC and is routinely used to simultaneously control the beam orbit, machine tune, and radial-loop adjusting the beam energy. The system handles the input of over 2'000 measurements, and controls the current in over 1'000 superconducting dipole correction magnets, over 200 quadrupole correction magnets as well as the RF frequency used to generate the electric field accelerating the particle beams. Recently, a new team was charged with maintaining, documenting and upgrading the software in order to meet the requirements for LHCs 2nd run. The team identified several requirements: 1) gather statistics on the relative offsets of the arrival of measurement data, 2) inspect RT I/O in a user-friendly way, 3) display summarized status information on the synchronism and content of the input data, 4) have the means to rapidly diagnose problems with the feedbacks during commissioning and operation in a non-intrusive way (i.e. without compromising the feedback systems real-time behavior).**

This paper documents the design, integration and use of the resulting monitoring suite for the LHC beam-based feedback systems. The set-up comprises a FESA-based (framework for real-time systems developed at CERN) real-time server and a JavaFX-based graphical interface integrated into CERNs operational software infrastructure. Concrete examples are given on how this system has contributed to a better understanding of the overall feedback behavior and aided in diagnosing operational problems. The paper will also summarize envisaged requirements for future releases.

I. Intro

The main source of disturbances on the orbit of the particles accelerated in the LHC is resulting, mainly, from quadrupole magnet misalignments causing deflections on the particles' trajectories. These are non-stationary perturbations, i.e. they depend on ground motion, earth tides and optics changes. Overall, this effect contains a mix of random and reproducible components and active feedback control is required to ensure that the particles remain contained inside the beam pipe. The beam orbit in the LHC is measured by more than 2'000 Beam Position Monitor (BPM) sensors [1] spread over two 27Km rings. Localised corrections of the beam orbit are performed by controlling the current in over 500 dipole magnets whereas the average radial position of the beam is controlled by adjusting the RF frequency of the electromagnetic cavities accelerating the beam.

Manuscript received June 2, 2016.

Diogo Alves (e-mail: Diogo.Alves@cern.ch), Stephen Jackson (e-mail: Stephen.Jackson@cern.ch) and Iason-Dimitrios Rodis (e-mail: Iason-Dimitrios.Rodis@cern.ch) are with the Beams Department, Beam Instrumentation Group, European Organization for Nuclear Research, CERN, F-01631 Cedex, France.

The focusing properties of the LHC beams is determined by its so-called focusing-defocusing lattice [2]. Beam particles in the presence of such a type of lattice describe what is known as betatron oscillations [2]. The number of oscillations describe by each particle in one complete revolution is know as the tune and it plays a crucial role as far as particle confinement is concerned [2]. Uncertainties in the machine tune occur during dynamic phases (particularly at injection) and whenever the energy and/or the optics change. In the LHC, tune feedback is particularly important during the first part of the energy ramp where transients may lead to significant particle losses. The four tune values (horizontal and vertical for both beams) are measured using up-to 12 tune measurement sensors [3] and are corrected by adjusting the current in over 250 quadrupole magnets.

The centralised feedback controller [4] (see Fig. 1) exchanges measurement and status information as well as actuator requests, in real-time, with a wide number of Front-End Computers (FECs), each managing several measurement or actuator devices. The real-time communication between the feedback system and these FECs is done using a switched ethernet-based network.

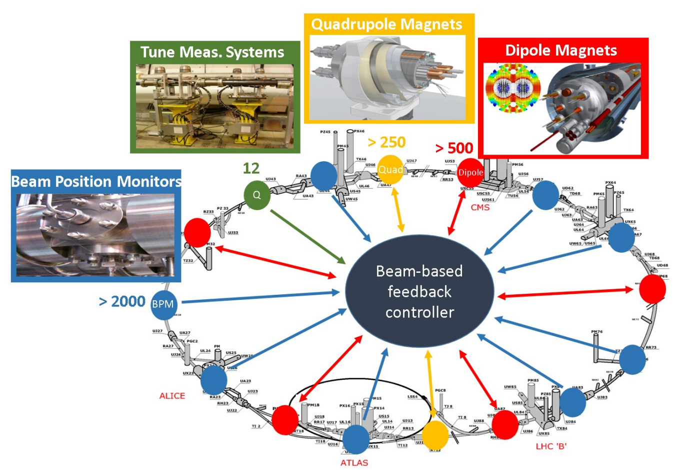

Fig. 1. LHC beam-based feedbacks overview

II. System Architecture

The beam-based feedback system in the LHC comprises two C++ servers [5]: a feedback controller running the algorithms for the control of the orbit, radial-loop and tune, and a service unit handling all activities related to monitoring, logging and proxying of the controller's settings and data (see Fig. 2). These servers run on 32GB Hewlett-Packard Proliant Gen8-based machines with 24 (hyper-threaded) cores running at 2.6GHz. The real-time communication between the servers relies on an object serialization mechanism based on ROOT (a scientific library developed at CERN) and transported using standard IP protocols over a private ethernet link. Both sensor measurements and actuator requests are transmitted using the UDP protocol over the LHC's technical network. The dominant input network traffic associated with the beam-based feedback control in the LHC is due to 25Hz beam position measurements synchronously sent by 67 FECs and adding up to a data rate of approximately 2MB/s. Each of these FECs can handle measurement data from up-to 36 sensor devices.

Recently, a new team was charged with maintaining, documenting and upgrading the software for the LHC beam-based feedbacks in order to meet the requirements for the LHC's 2nd run. This included the porting from 32 to 64-bits and from FESA [6] version 2 to FESA version 3 as well as the improvement of the overall system's stability. Simultaneously, the team had to ensure that the compatibility of the system with the LHC's operational procedures and its previous control performance were not impacted by the changes. On these points in particular, a dedicated testbed system and Java-based framework [7] were developed to validate, from an operational point-of-view, the software changes.

Due to its complexity and importance, monitoring and diagnosis tools are crucial for testing and operating such a system. Although there already existed high-level tools for monitoring processed (i.e. decimated) data, the team identified a deficiency in tools to perform the low-level inspection of the real-time input/output data. In particular, a tool that would allow to: 1) gather statistics on the relative offsets of the arrival of measurement data, 2) inspect RT I/O in a user-friendly way, 3) display summarized status information on the synchronism and content of the input data, 4) have the means to rapidly diagnose problems with the feedbacks during commissioning and operation in a non-intrusive way (i.e. without compromising the feedback system's real-time behaviour).

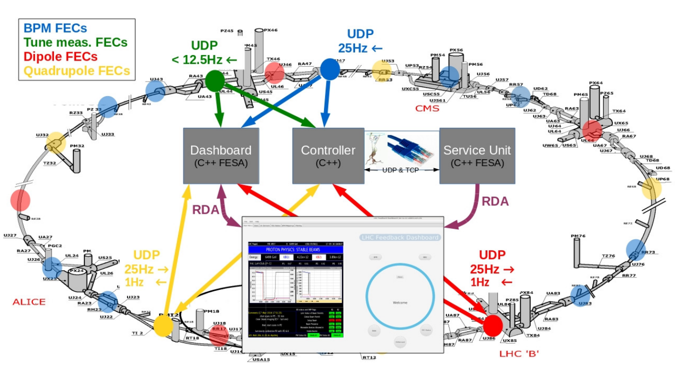

Fig. 2. System architecture

With these requirements in mind, a new monitoring suite for the LHC beam-based feedbacks was developed. This suite comprises two pieces of software: a low-level C++ FESA-based server and a Graphical User Interface (GUI), see Fig. 2. The C++ server's main responsibilities are to monitor and provide statistics information on the synchronism of the arrival of the real-time input data as well as forwarding the raw data requested by the GUI application. This server receives a copy of all real-time input/output data used by the feedback controller. Its most demanding processing task is to read each BPM UDP packet, calculate statistics on the latency and jitter of the arrival, perform checks, publish (in real-time) a subset of position measurements to be used by other systems and buffer the raw data to be provided (on-demand) to the GUI application. The GUI is one of the first JavaFX-based[1] applications developed by the software section of the beam instrumentation group following the official recommendation for moving from the deprecated Java-Swing[2] framework. It is used to process and display, in an intuitive manner, the data coming from the low-level server. Its most demanding processing task is to perform checks on both the statistics data calculated by the low-level server, and the decimated position data published by the feedback service unit, and display summarised status information. The fastest refresh rate in the GUI is 1Hz. A prototype dataviewer component has also been developed to provide data plotting capabilities in the application. This component was developed using the free *plotly*[3] javascript library for charts, *Jetty*[4] (a light-weight Java-based web server) and *WebSockets*[5]. It supports typical 2D (line, scatter, area, contour, etc) and 3D plots as well as crucial interactive capabilities such as zooming and information on hover. Although JavaFX provides native charting components, it does not offer the essential interactive functionalities that can be found, e.g., in *plotly*. In an effort to provide a common generic JavaFX-based dataviewer to be used by GUIs in the group, it is envisaged at a later stage, to benchmark this approach with another prototype using the native JavaFX charting components. Using the GUI application, is possible to plot raw sensor data as well as the decimated data published by both the low-level server and the feedback service unit.

III. Functionality

Three main functionalities are presently being provided by this monitoring suite. The first is the real-time display of the low-level status of the various BPM FECs, see Fig. 3. Two different status categories are separately treated: the status related to the synchronism of the arrival of the measurement data and the status related to the actual content of the measurement data. Visually, statuses are represented by a colour code (green, yellow, orange and red in increasing order of severity condition). Each colour is the result of a series of tests performed on data received, via Remote Data Access[6] (RDA), from either the Dashboard C++ server or the feedback service unit. A condensed status display, where all

[1]http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm
[2]http://docs.oracle.com/javase/tutorial/uiswing/start/about.html
[3]https://plot.ly/javascript
[4]http://www.eclipse.org/jetty
[5]https://www.w3.org/TR/2009/WD-websockets-20091222
[6]Data transfer protocol developed at CERN.

BPM FECs are grouped according to the LHC octant where they are physically placed, has been implemented. Whenever one finds a suspicious occurrence on a particular BPM FEC, one can easily navigate down to the particular test and BPM device in question.

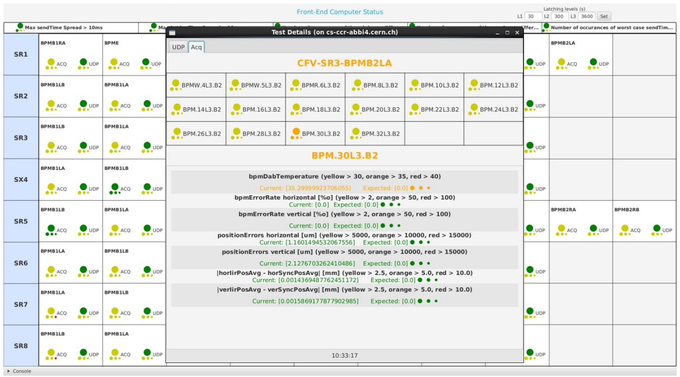

Fig. 3. BPM crate status tab

The second main functionality is the capability, from the GUI, to visualise the content of the UDP packets coming from the BPM FECs. This functionality is accessed via a dedicated tab in the GUI application which allows for combined filtering on LHC octant and FEC name as well as the specification of the desired number of sampled UDP packets. The data itself can be displayed as a table and as a 2D plot, see Fig. 4.

The third main functionality provided by the GUI is found in a dedicated tab which allows the user to select and plot the data fields available via the RDA subscriptions, namely coming from the dashboard C++ server and the feedback service unit.

Other functionalities provided by this newly developed monitoring suite include: displaying BPM sensor calibrations, displaying the operational state (enabled/disabled) of the BPMs, cross-checking BPM device mapping, automatic email notifications whenever UDP packets stop being received by the low-level server and automatic report generation (typically once a day) on identified error conditions in order to monitor potential system degradation over time.

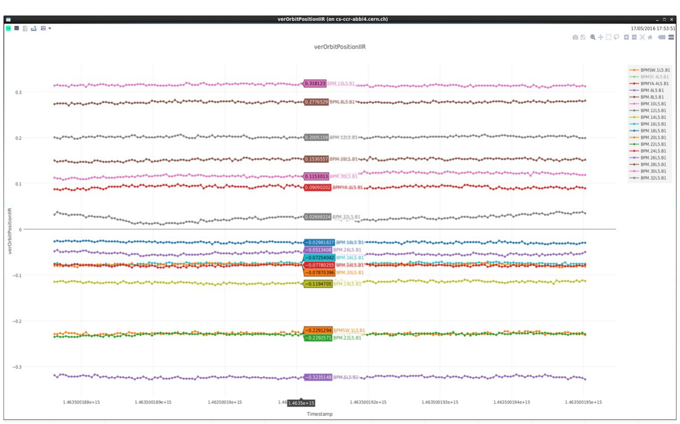

Fig. 4. 2D plot of raw BPM data

IV. Conclusions

Since it has been deployed, the system has been greatly used not only by the software team that designed and developed it, but also by instrumentation experts in seek of a tool to assist them during commissioning and operational maintenance. This system has helped diagnose, rapidly and effectively, several problems during the LHC's restart/commissioning and operation. For example, it was instrumental in tracing back a hardware problem which occasionally affected the time-stamp of the orbit measurement. The consequence was to trigger occasional, and at a first glance non-reproducible, beam-dumps on checks performed by the software interlock system. Other problems included diagnosing problems related to invalidated measurements due to data sanity issues which was bearing a negative impact on the effective observability of the system with respect to the control of the orbit. It is routinely used to double check problems related to the loss of measurements which occasionally occur when measurement FECs cease to transmit their UDP packets. In addition to the monitoring functionality, the system has also been used as a fast beam trajectory data source using CERN's standard RDA protocol. It has been used for quickly prototyping an algorithm based on the fast position measurements for correcting intensity measurements. Furthermore, it has also been used for cross-checking position oscillations observed with the synchrotron radiation telescope.

In view of its demonstrated usefulness, the system will be extended in order to be able to also monitor tune measurement data and actuator requests as well as their voltage and current readbacks.

References

[1] O. R. Jones, *LHC Beam Instrumentation*, PAC 2007, Albuquerque, New Mexico, USA.
[2] Edmund Wilson, *The Physics Of Particle Accelerators*, Oxford University Press, first edition, 2001.
[3] M. Gasior and O. R. Jones, *High Sensitivity Tune Measurement By Direct Diode Detection*, PAC 2005, Lyon, France.
[4] R. Steinhagen et al., *Real-Time Beam Control at the LHC*, PAC 2011.
[5] L. K. Jensen et al., *Software Architecture for the LHC Beam-Based Feedback System at CERN*, ICALEPCS 2013.
[6] M. Arruat et al., *Front-End Software Architecture*, ICALEPCS 2007.
[7] S. Jackson et al., *Testing Framework for the LHC Beam-Based Feedback System*, ICALEPCS 2015.