ONLINE COUPLING MEASUREMENT AND CORRECTION THROUGHOUT THE LHC CYCLE

A. Calia, K. Fuchsberger, M. Gabriel, G.-H. Hemelsoet, M. Hostettler, M. Hruska, D. Jacquet, M. Soderen, T. Persson, D. Valuch, CERN, Geneva, Switzerland

Abstract

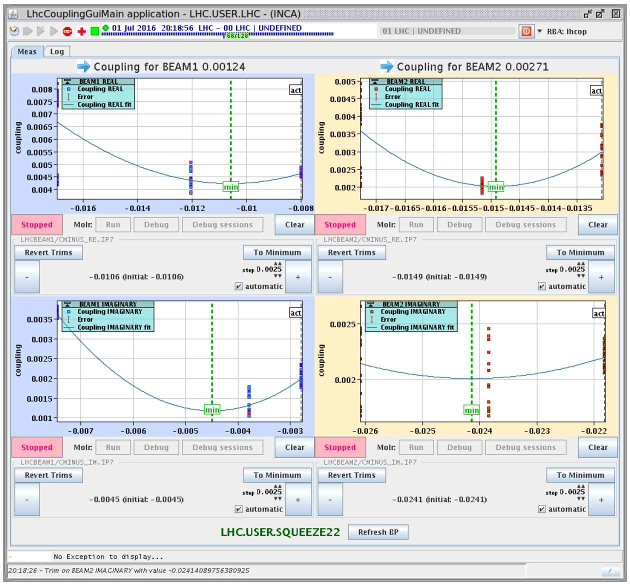
With high intensity beams, a precise measurement and effective correction of the betatron coupling is essential for the performance of the Large Hadron Collider (LHC). In order to measure this parameter, the LHC transverse damper (ADT), used as an AC dipole, will provide the necessary beam excitation. The beam oscillations will be recorded by the Beam Position Monitors and transmitted to dedicated analysis software. We set up the project with a 3-layer software architecture: The central part is a Java server, orchestrating the different actors: The Graphical User Interface, the control and triggering of the ADT AC dipole, the BPMs, the oscillation analysis (partly in Python), and finally the transmission of the correction values. The whole system was developed in a team using Scrum, an iterative and incremental agile software development framework. In this paper we present an overview of the system, experience from machine development and commissioning as well as how scrum helped us to achieve our goals. Improvement and re-use of the architecture with a nice decoupling between data acquisition and data analysis are also briefly discussed.

INTRODUCTION

A proper tune control requires a machine coupling which is significantly smaller than the tune separation. $|C-|$, the complex coupling coefficient corresponds physically to closest approach of the 2 tunes, Q_x and Q_y. For the LHC this requirement yields an operational tolerance for the global machine coupling of $|C-| \ll 0.03$ for injection optics and $|C-| \ll 0.003$ for collision optics [1,2]. The tune feedback, an essential feature for the beam stability, requires that the coupling is as low as possible. Otherwise it is impossible to correct the horizontal and vertical tune independently as a correction in one plane will also influence on the other. The transverse coupling has furthermore been linked to cause instabilities and the reduction of dynamic aperture [3].

MEASURING TRANSVERSE COUPLING

The method to measure transverse coupling described in this article is to excite the beam with the transverse damper (ADT) used as an AC dipole and record the beam position during 6600 turns. This leads to a coherent beam motion close to the betatron frequency. Unlike kickers, an AC dipole excitation can produce a sustained coherent motion with negligible emittance growth. For that reason, this became the preferred mode of excitation for such kind of measurements in the LHC. Turn-by-turn position data of the excited motion from beam position monitors (BPMs) allows prompt measurements of optics parameters. Coupling Resonance Driving Terms (CRDT) due to skew quadrupole fields can be determined from corresponding spectral components of the turn by-turn position [4–6]. The benefit of using the ADT to excite the beam is that it enables a well controlled excitation of an individual bunch(es), while the traditional AC dipole would excite all bunches in the machine [7, 8].



```
No Exception to display...
20:18:26 - Trim on BEAM2 IMAGINARY with value -0.02414089756380925
```

Figure 1: Old manual application.

MOTIVATION FOR NEW SOFTWARE

Before the described software was put in operation, there were essentially 3 methods to measure the coupling in the LHC:

- The first one was a lengthy process of trial with the 2 knobs controlling the coupling correction (real and imaginary part) trying to find a minimum in $|C-|$. This has to be done for the 2 beams individually (Fig. 1). Also, the operational measurement of $|C-|$ is not reliable in all machine configurations.
- In 2016 operation, a software taking advantage of the injection oscillations was used. This approach was intrinsically limited to corrections at injection energy and required a dump and re-injection of low-intensity bunch for each measurement.
- An expert tool was used by the optics team to measure optic parameters, including the coupling, using the LHC AC dipole. However, for machine protection reasons, powering the AC dipole is restricted to

low-intensity beams and thus was only used during dedicated optics measurement machine cycles in the beginning of the run.

None of those methods was perfect, essentially because they required too much time or were too complex to use. The essential motivation was to correct efficiently the coupling throughout the full LHC machine cycle in a minimum of time.

SOFTWARE DESIGN

The aim of this new software is to simplify the measurement process by allowing the different actors to work together under the supervision of a central server. Different technologies and languages were used. In particular the top layer of the LHC control system is based on Java, while the existing data analysis scripts for LHC optics measurement and correction [9] were implemented in Python (Fig. 2).

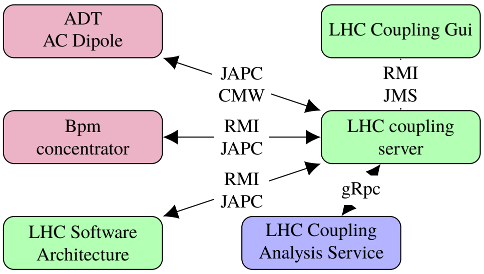


Figure 2: overview of the new coupling software architecture (green: Java, blue: Python red: Fesa)

MEASUREMENT FLOW

The system operates as shown in Fig. 3:

- The operator enters the desired parameters for the measurement (the natural tune, the offset between the tune and the excitation, the position of the bunch to be excited and the excitation amplitude) and triggers the measurement.
- The timing system then triggers the ADT excitation and the recording of the BPMs.
- When the BPM acquisition finishes, the data is sent to the Python analysis service to compute the coupling corrections.
- The Python analysis service returns the measured coupling and proposed corrections for both beams. This data is forwarded by the Java server to the GUI.
- The operator can decide to apply the proposed correction (or a modification thereof) by trimming the coupling correction knobs.

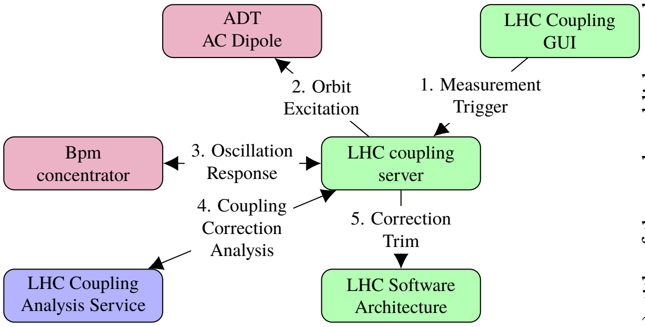
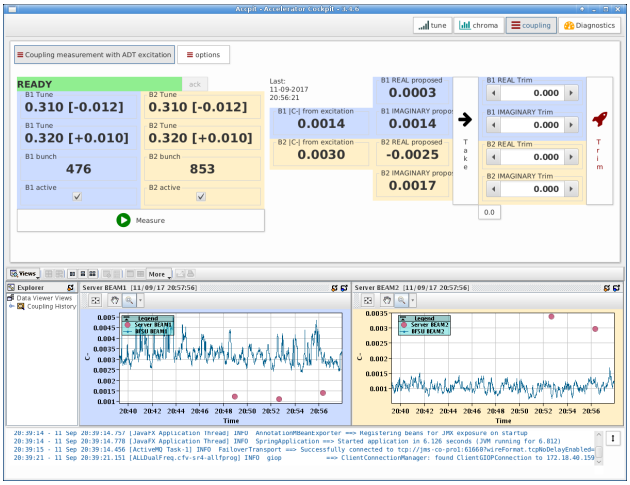


Figure 3: Overview of measurement and analysis process.

SOFTWARE COMPONENTS

Graphical User Interface

The GUI is used to interact with the operations crew. The natural tunes, the tune offsets, the bunch to excite and the amplitude are set via the application. The bunches present in the machine are either set automatically or overwritten by the user. After this input is provided, the user can trigger a measurement. The progression of the measurement and the proposed corrections are shown afterwards. Finally, an interface to send the corrections to the hardware is also available. The GUI uses small GUI-composition framework (*MiniFx*), developed by the LHC operational software team (Fig. 4).



READY

B1 Tune 0.310 [-0.012]
B2 Tune 0.310 [-0.012]
B1 Tune 0.320 [+0.010]
B2 Tune 0.320 [+0.010]
B1 bunch 476
B2 bunch 853
B1 active ☑
B2 active ☑

Last:
11-09-2017
20:56:21

B1 |C-| from excitation 0.0014
B2 |C-| from excitation 0.0030
B1 REAL proposed 0.0003
B1 IMAGINARY propos 0.0014
B2 REAL proposed -0.0025
B2 IMAGINARY propos 0.0017
B1 REAL Trim 0.000
B1 IMAGINARY Trim 0.000
B2 REAL Trim 0.000
B2 IMAGINARY Trim 0.000
0.0

Server BEAM1 [11/09/17 20:57:56]
Server BEAM2 [11/09/17 20:57:56]

```
20:39:14 - 11 Sep 20:39:14.757 [JavaFX Application Thread] INFO  AnnotationMBeanExporter ==> Registering beans for JMX exposure on startup
20:39:14 - 11 Sep 20:39:14.778 [JavaFX Application Thread] INFO  SpringApplication ==> Started application in 6.126 seconds (JVM running for 6.812)
20:39:15 - 11 Sep 20:39:14.456 [ActiveMQ Task-1] INFO  FailoverTransport ==> Successfully connected to tcp://jms-co-pro1:61660?wireFormat.tcpNoDelayEnabled=
20:39:21 - 11 Sep 20:39:21.151 [ALLDualFreq.cfv-sr4-allfprog] INFO  giop          ==> ClientConnectionManager: found ClientGIOPConnection to 172.18.40.159
```

Figure 4: Graphical User Interface

LHC Coupling Server

The server orchestrates the various actions: it avoids that multiple measurements are launched at the same time and keeps track of the progress of each measurement. It uses timing events to trigger different actions. The server can be run in "demo" mode to allow e.g. testing the GUI part or running the analysis part on historical data without actually accessing the LHC hardware. To integrate the various dissimilar modules and systems, it uses a variety of different communication technologies.

Beam Position Monitors

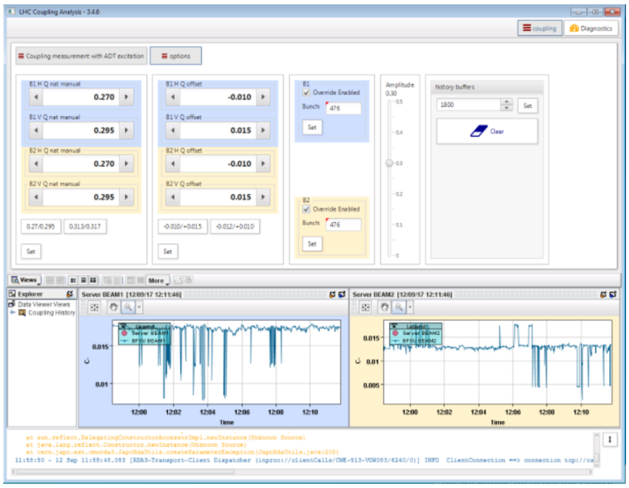
The system currently uses the standard Beam Position Monitor system to measure the excitation response. In the future it could alternatively use the LHC Diode ORbit and OScillation (DOROS) system which provides beam position readings with a better resolution [10].

LHC Coupling Analysis Service

This Python service communicates with the LHC coupling server using gRPC. An analysis is triggered by the coupling server upon receiving the BPM data. A noise reduction technique based on Singular Value Decomposition where only the the most fundamental modes for the motion is applied [11]. A Fourier decomposition of the cleaned turn-by-turn data is then used to obtain the main frequency and phase of the motion at each BPM. Based on this, the coupling strength at each measurement point can be reconstructed. In order to correct the coupling a model which predicts the influence on each coupling knob is created. Using this model in combination with the measurement, the service calculates the coupling corrections, which are returned to the Java server.

LHC Software Architecture (LSA)

The CERN standard frameworks are used to access the equipment: A new Fesa 3 [12] class was designed to control the transverse damper excitation subsystem (Fig. 5), i.e., the natural tune, the tune offset, the position of the bunch to be excited and the excitation amplitude. The standard timing tables to send event to the various equipment and the standard LSA trim to send the correction values to the power supplies are used.


LHC Coupling Analysis - 3.4.6

B1 H Q nat manual 0.270
B1 V Q nat manual 0.295
B2 H Q nat manual 0.270
B2 V Q nat manual 0.295
0.27/0.295 0.313/0.317

B1 H Q offset -0.010
B1 V Q offset 0.015
B2 H Q offset -0.010
B2 V Q offset 0.015
-0.010/+0.015 -0.012/+0.010

B1 ☑ Override Enabled Bunch 476
B2 ☑ Override Enabled Bunch 476

Amplitude 0.30

history buffers 1800

Server BEAM1 [12/09/17 12:11:46]
Server BEAM2 [12/09/17 12:11:46]

```
    at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(Unknown Source)
    at java.lang.reflect.Constructor.newInstance(Unknown Source)
    at cern.japc.ext.cmwrda3.JapcRdaUtils.createParameterException(JapcRdaUtils.java:205)
11:58:50 - 12 Sep 11:58:48.083 [RDA3-Transport-Client Dispatcher (inproc://clientCalls/CWE-513-VOM083/6240/0)] INFO  ClientConnection ==> connection tcp://cs
```

Figure 5: Settings for the transverse damper (ADT AC dipole).

Technologies

Docker Image Since we were obliged to use Python for part of the system (because code for the relevant algorithms was already available in this language), we had to find a good deployment strategy for Python. It turned out that dependency management approaches mostly act on the Python distribution installed on the operation system and not on a per-application basis. This implies that if e.g. running two Python applications that potentially require libraries in different versions would be impossible on the same host. The way around this was finally to package the Python application, together with the correct Python distribution and all the required libraries in a docker image and run everything in a container [13]. By using Docker, flexibility is given to both the Python script owner and the system engineer to change e.g their Python version independently from each other. This avoids problems resulting from uncontrolled incompatible changes in the libraries while using a well known, easy to install and well supported product with a lighter load than with virtual machines.

gRPC protocol One of the challenges of this project is the communication channel between python and java. When investigating different solutions to interface the python scripts with java, gRPC (Google Remote Procedure Call, [14]) proved to be the most appropriate for our purpose: It supports a wide variety of languages, asynchronous communication and is fully implemented on top of http. Therefore it is prepared for rerouting and load balancing. In our case, we implemented a gRPC server in Python (the coupling analysis service) and a client in Java (the coupling server).

Spring and Software reuse The described coupling measurement system is part of a larger software suite, allowing the control of the main parameters of the machine (tune, chromaticity and coupling). It was easy and fast to set up as it relies on other basic bricks already written by the op-software team, for example the GUI composition library MiniFx, streaming pool [15] and the tensorics [16] library. The use of spring [17] also allows easy reconfiguring of the components so that one can reuse the same set of objects for different usage (e.g. the same widgets for chroma and tune). The dependency injection mechanism provided by spring allows us to better decouple the classes make them modular and write tests much more easily. Having each element already tested reinforces the cohesion of each part and makes the integration easier.

WORKING IN A TEAM

Since this project involves four CERN groups working together (BE-CO-APS, BE-OP-LHC, BE-ABP-LAT and BE-RF-FB), it was highly desirable to use agile techniques. In particular, the scrum framework [18] was applied to manage this project. The scrum framework (Fig. 6) can be described as following:

- A product owner creates a prioritized wish list (requirements), called a product backlog.
- During sprint planning meeting, the team pulls a small chunk from the top of that wish list, a sprint backlog, and decides how to implement those pieces.
- The team has a certain amount of time — a sprint (usually two to four weeks) — to complete its work, but it meets each day to assess its progress (daily Scrum).
- Along the way, the ScrumMaster keeps the team focused on its goal.
- At the end of the sprint, the work should be potentially shippable: ready to hand to a customer, put on a store shelf, or shown to a stakeholder.
- The sprint ends with a sprint review (demo with users) and a sprint retrospective (how one can improve the process).
- When the next sprint begins, the team chooses another chunk of the product backlog and repeats the process.

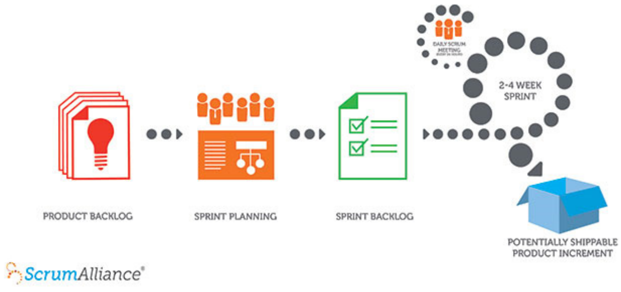


Figure 6: Scrum process graph [19].

Unlike the usual project management where a developer could end up working alone on a big project, working as a team was essential for the project to be realized faster and maintained on the long term. The other benefit of this framework are an enhanced collaboration betweens groups and persons, an increased flexibility to deal with the last minute changes and a working piece of software at the end of each iteration. Pair programming was also valuable as it spreads the knowledge inside the software team. The shutdown period was the best time to launch such project. Sill, not all the groups have the same schedule and sometimes it was difficult to find time to work all together. By applying the sprint concept, we were able to concentrate the tasks and thus everybody feels responsible for the sprint goal. At the end of each sprint there was a retrospective allowing team members to change the working habits and maximizing the team performance. This framework was also very efficient in taking into account the inevitable last minute changes and the daily scrum avoids that people are blocked for a longer period of time while waiting for dependent changes. As this software was written in a team and has reasonably well covered with unit tests, it is less of a problem when a developer quits as the rest of the team can take over. This is particularly important within the operations group, as some application are already orphan, inherited from people who left CERN, which we want definitely not increase.

FIRST OPERATIONAL EXPERIENCE

This software is operational since August 2017 (LHC fill 6100) and has been used in every fill since. Unlike the previous method it does not require a lengthy process of trims and measurements but can derive a correction directly from a single measurement. Further, the system allows to measure in any point of the cycle whereas the injection coupling measurement was limited to the injection process. Currently, due to machine protection restrictions, it is allowed to excite only one bunch in the machine and the damper software will reject dangerous values in the excitation settings.

As illustrated in Fig. 7, the new software allows a coupling correction down to the $|C-| \approx 0.001$ level for most fills, whereas the uncorrected coupling would be up to a factor of 10 higher. This is significantly better than the minimum requirement given in the LHC design report, and provides additional margin for beam stability.

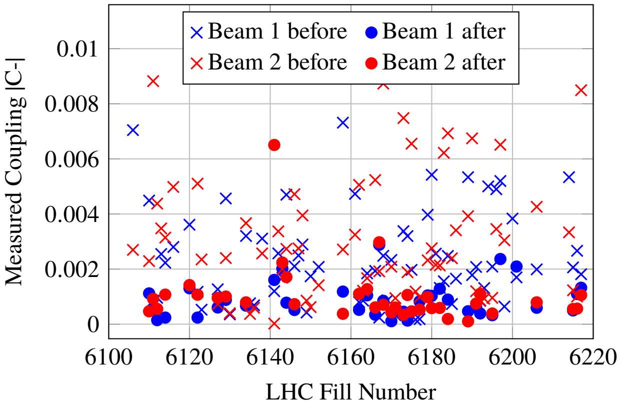


Figure 7: $|C-|$ before and after correction, as measured by the new coupling measurement tool.

The difference of the $|C-|$ values before and after the correction allows to estimate the convergence of the correction algorithm (Fig. 8). In ~90% of the fills, the tool improved the coupling correction by decreasing the $|C-|$. From the cases where the correction was not improved, and from the level we can correct the coupling to, we can estimate the precision of the measurement to ~0.001 in $|C-|$.

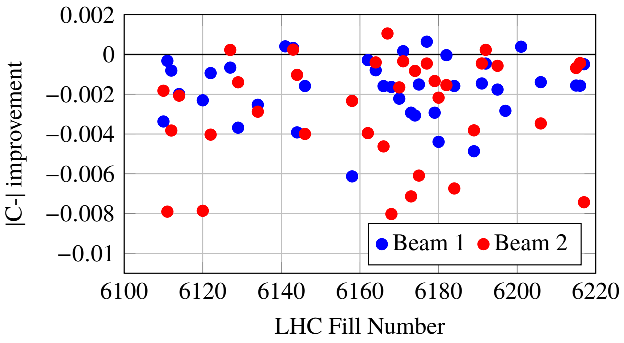


Figure 8: Difference of $|C-|$ before and after correction, measured by the new coupling measurement tool. Negative values correspond to coupling improvement, positive to degradation.

CONCLUSION

Thanks to the described new software, the coupling is now corrected efficiently all along the cycle without requiring the help of optics specialists. It reduces also the measurement time taken at each injection, improving the global performance of the LHC machine. The team members found it interesting - though challenging - to choose and learn those new technologies which will be highly valuable for future software developments within operations. Substantial development time was saved using the existing and already tested software and, since the knowledge is shared among the team members, it will allow for faster development of applications. Concurrently with the Luminosity Control [20], it was the first time the scrum framework was used in the team for a real project and it helped greatly to improve the way code is written. The main benefits of this framework are an increased collaboration, a continuous training, an increased attention to the details so that the code remains readable and testable, and a continuous improvement of the working habits.

ACKNOWLEDGMENT

The authors would like to express their thanks to the management for good collaboration within the BE groups and for the nice working environment they had for this project development. We would also like to thank the support we had for the training required for this project. Last but not least, we are thankful to everybody in the operations team who directly and indirectly helped in the design and improvements of the accelerator cockpit application by their constructive criticisms and remarks. Similarly thanks to the ABP and RF teams who provided valuable inputs to this tool.

REFERENCES

[1] M. Benedikt, P. Collier, V. Mertens, J. Poole, and K. Schindl, Eds., “LHC Design Report”, 2004, doi:10.5170/CERN-2004-003-V-3

[2] L.R. Carver *et al.*, “Destabilizing effect of linear coupling in the LHC”, in Pro. IPAC’17, Copenhagen, Denmark, May 2017, pp. 3791–3794, doi:10.18429/JACoW-IPAC2017-THPAB040.

[3] E.H. Maclean, F. Carlier, M. Giovannozzi, T.H.B. Persson, and R. Tomás, “Effect of Linear Coupling on Nonlinear Observables at the LHC”, in Pro. IPAC’17, Copenhagen, Denmark, May 2017, pp. 3151–3154, doi:10.18429/JACoW-IPAC2017-WEPIK092

[4] R. Tomás,T. Persson, and E. Maclean, “Amplitude dependent closest tune approach”, *Phys. Rev. Accel. Beams* vol. 19, p. 071003, 2016, doi:10.1103/PhysRevAccelBeams.19.071003

[5] T. Persson and R. “Improved control of the betatron coupling in the Large Hadron Collider”, *Phys. Rev. Spec. Top. Accel. Beams*, vol. 17, p. 051004, 2014, doi:10.1103/PhysRevSTAB.17.051004

[6] T. Persson, Y. Inntjore Levinsen, E.H. Maclean, and R. Tomás, “Chromatic coupling correction in the Large Hadron Collider”, *Phys. Rev. Spec. Top. Accel. Beams*, vol. 16, p. 081003, 2013, doi:10.1103/PhysRevSTAB.16.081003

[7] R. Miyamoto, R. Calaga, M. Aiba, R. Tomás, and G. Vanbavinckhove, “Measurement of coupling resonance driving terms in the LHC with AC Dipoles”, in *Proc. 2nd International Particle Accelerator Conference (IPAC’11)*, San Sebastian, Spain, Sep. 2011, pp. 2067–2069, https://cds.cern.ch/record/1382073

[8] D. Valuch *et al.*, “ADT and ObsBox in 2016”, presented at the 7th Evian Workshop, France, Dec. 2016.

[9] Beta-Beat, https://github.com/pylhc/Beta-Beat.src

[10] T. Persson *et al.*, “Experience with DOROS BPMs for coupling measurement and correction”, in *Proc. 7th International Particle Accelerator Conference (IPAC’16)*, Busan, Korea, May 2016, pp. 303–305, doi:10.18429/JACoW-IPAC2016-MOPMR029

[11] R. Calaga and R. Tomas, “Statistical analysis of RHIC beam position monitors performance”, *Phys. Rev. ST Accel. Beams*, vol. 7, p. 042801, 2004, doi:10.1103/PhysRevSTAB.7.042801

[12] M. Arruat *et al.*, “Front-end software architecture”, in *Proc. ICALEPCS’07*, Knoxville, TN, USA, Oct. 2007, pp. 310–312.

[13] Docker, https://www.docker.com

[14] GRPC, https://grpc.io

[15] A. Calia *et al.*, “Streaming Pool - Managing long-living reactive streams for Java”, presented at ICALEPCS’17, Barcelona, Spain, Oct. 2017, paper THPHA176.

[16] K. Fuchsberger *et al.*, “Tensorics - a Java library for manipulating multi-dimensional data with pleasure”, presented at ICALEPCS’17, Barcelona, Spain, Oct. 2017, paper THPHA177.

[17] Spring, https://spring.io/

[18] Scrum, https://en.wikipedia.org/wiki/Scrum_(software_development)

[19] Scrum Alliance, https://www.scrumalliance.org/why-scrum

[20] M. Hostettler *et al.*, “Online luminosity control and steering at the LHC” presented at ICALEPCS’17, Barcelona, Spain, Oct. 2017, paper TUSH201.