# CLOSED ORBIT FEEDBACK FOR FAIR - PROTOTYPE TESTS AT SIS18

B. R. Schlei*, H. Liebermann, D. Ondreka, P. J. Spiller, R. J. Steinhagen
GSI Helmholtzzentrum für Schwerionenforschung GmbH, Darmstadt, Germany

*Abstract*

A new steering software for cycle-to-cycle closed orbit as well as trajectory control is currently under development for FAIR's planned control system. It has been successfully tested with beam at the SIS18 in 2016. COAT (i.e., Controlling Orbits And Trajectories) has been realized as a distributed, Java-based application. It consists of a background daemon process that handles the actual beam-based feedback logic, and independent clients that provide visualization and various user-interaction capabilities. Built on top of the LSA settings management system, code-shared and also used at CERN, the system is kept generic. Furthermore, it is designed to support multiple accelerators, transfer lines and users in parallel. In particular, it can handle continuously changing optics and other in advance known changing beam parameters. The COAT computer program is part of a set of newly developed beam-based feedback tools[1] for FAIR. Preliminary results of our proof-of-concept prototype studies indicate, e.g., in view of the observed SIS18 machine reproducibility, that such a cycle-to-cycle feedback control scheme may be adequate also for the other FAIR accelerators and transfer lines.

## INTRODUCTION

In addition to the beam position monitors (BPM) of the existing GSI facility, a new common data acquisition system is planned for the new FAIR synchrotrons and its high energy beam transport lines (HEBT) [1]. Eventually, all accelerators and the HEBT of both GSI and FAIR will be controlled through an adoption of the CERN LSA [2] setting generation system [3]. In order to provide a closed feedback system for these facilities, a new steering software named COAT has been developed. In 2016, its prototype has been tested at SIS18 with a $^{124}Xe^{43}$ beam.

## COAT - CONTROLLING ORBITS AND TRAJECTORIES

COAT has been realised as a distributed software application (*cf.*, Fig. 1). In particular, it is based on the Java Spring RMI framework [4, 5]. Its various BPM data acquisition services and monitoring clients (*cf.*, e.g., Fig. 2) may therefore interact while they run on different computer systems simultaneously. Figs. 2–7 demonstrate that COAT can actually handle continuously changing optics while providing SVD-based [6, 7] closed orbit corrections.

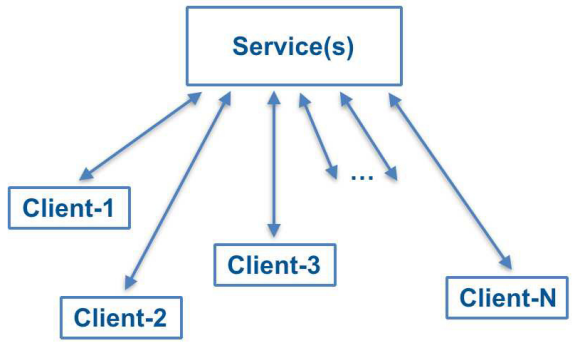


Figure 1: Design scheme of a distributed application with one or more services and a finite number, $N$, of clients.

## CONCLUSION

Although the automatic performance of a foreseen PI controller (*cf.*, e.g., Ref. [8]) could not be tested yet, we are confident that the current COAT prototype will handle both cycle-to-cycle closed orbit and trajectory control eventually quite adequately.

## ACKNOWLEDGEMENT

B. R. Schlei is indebted to J. Wenninger of CERN for valuable insights into his steering software YASP [9].

## REFERENCES

[1] R. Singh et al., "Benchmarking of DAQ system for FAIR beam position monitors", GSI Scientific Report 2015 (2016) p. 361, doi: 10.15120/GR-2016-1.

[2] G. Kruk et. al, "LHC Software Architecture LSA – Evolution Toward LHC Beam Commissioning", ICALEPCS'07.

[3] D. Ondreka et al., "Project Status of the New Setting Generation System for GSI and FAIR", GSI Scientific Report 2012 (2013) p. 254, PHN-FAIR-28.

[4] C. Walls, *Spring in Action*, Manning 2014.

[5] W. Grosso, *Java RMI*, O'Reilly & Associates 2001.

[6] M. G. Minty, F. Zimmermann, *Measurement and Control of Charged Particle Beams*, Springer 2011.

[7] G. H. Golub, C. F. Van Loan, *Matrix Computations*, J. Hopkins Uni. Press 2013.

[8] J. J. DiStefano, *Schaum's Outline of Feedback and Control Systems*, Mcgraw-Hill Education Ltd 2013.

[9] J. Wenninger, "YASP Steering Program User Guide"; for more detail, *cf.*, `http://jwenning.web.cern.ch/jwenning/documents/YASP/YASP-user-guide.pdf`

* b.schlei@gsi.de
[1] see separate contribution by R. J. Steinhagen et al.

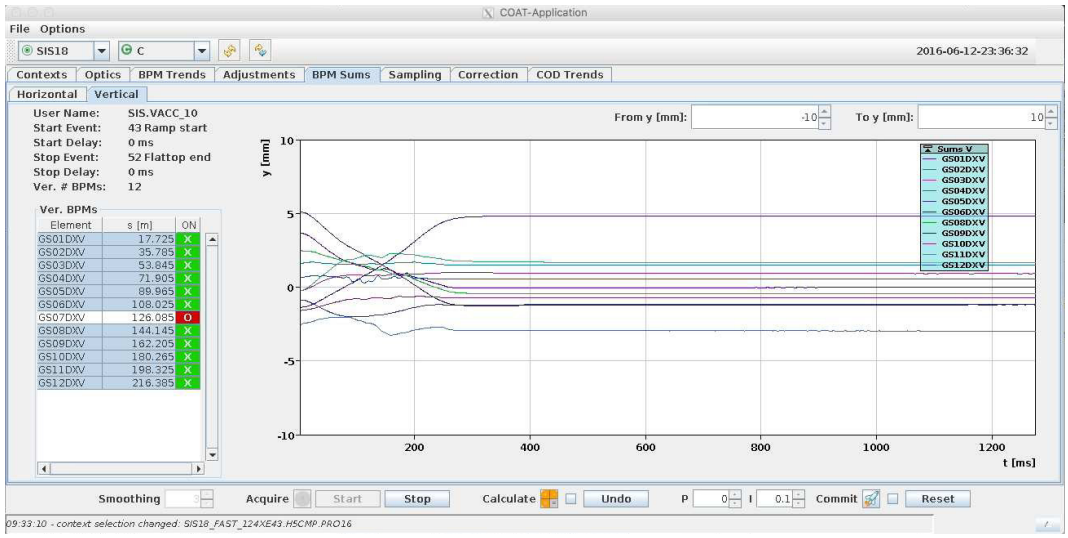


Figure 2: COAT Client, showing vertical BPM sums signals before correction. Note that GS07DX signals were ignored.

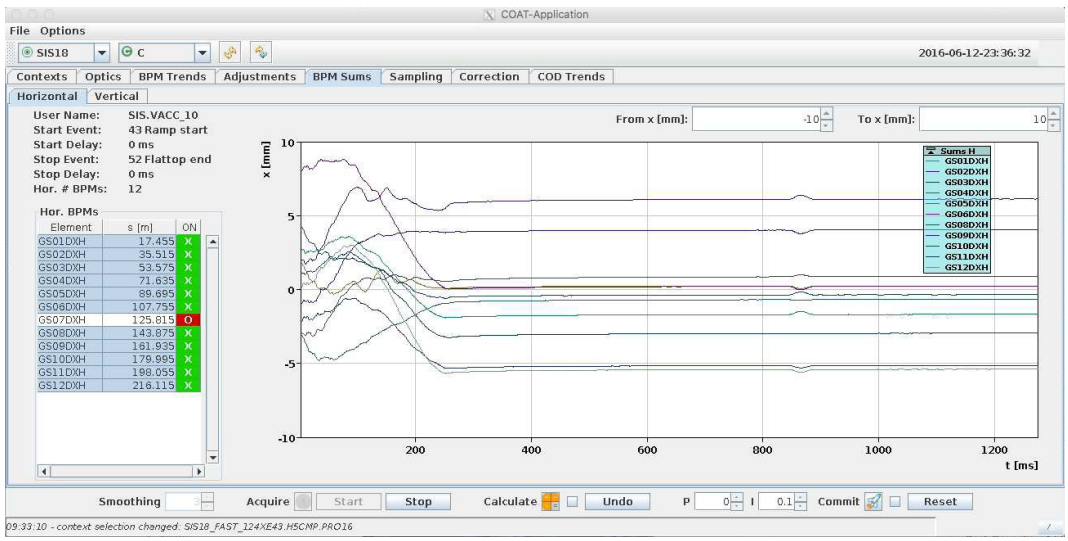


Figure 3: COAT Client, showing horizontal BPM sums signals before correction, while not accounting for dispersion.

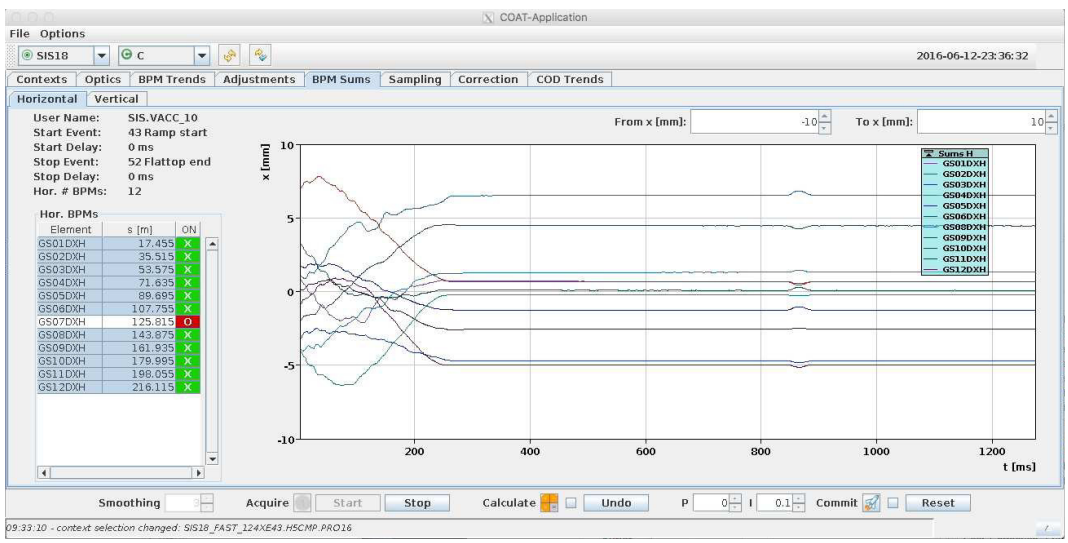


Figure 4: COAT Client, showing horizontal BPM sums signals before correction, with theoretical dispersion subtracted.




ISBN 978-3-95450-182-3

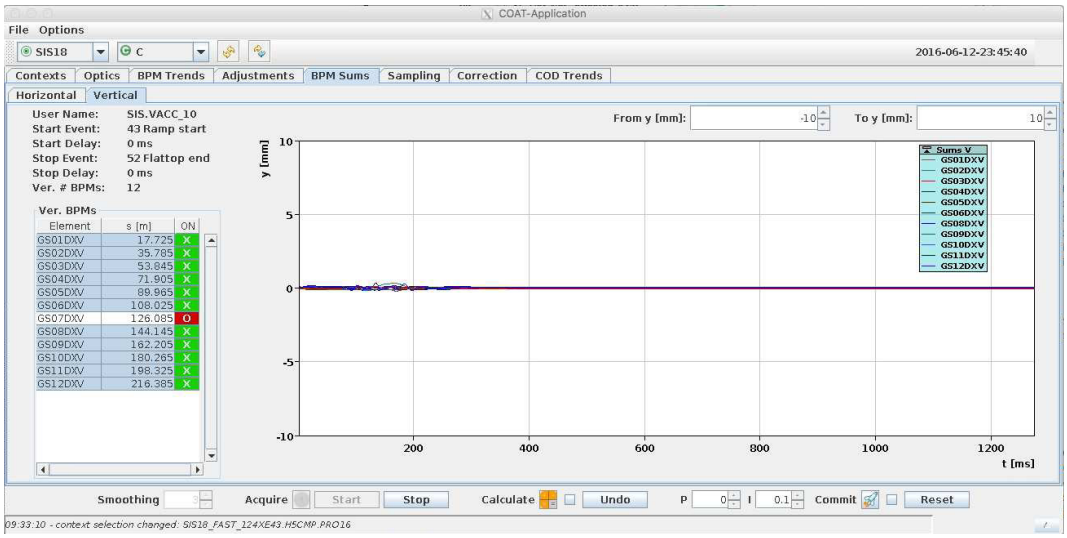


Figure 5: COAT Client, showing vertical BPM sums signals after correction. Note that GS07DX signals were ignored.

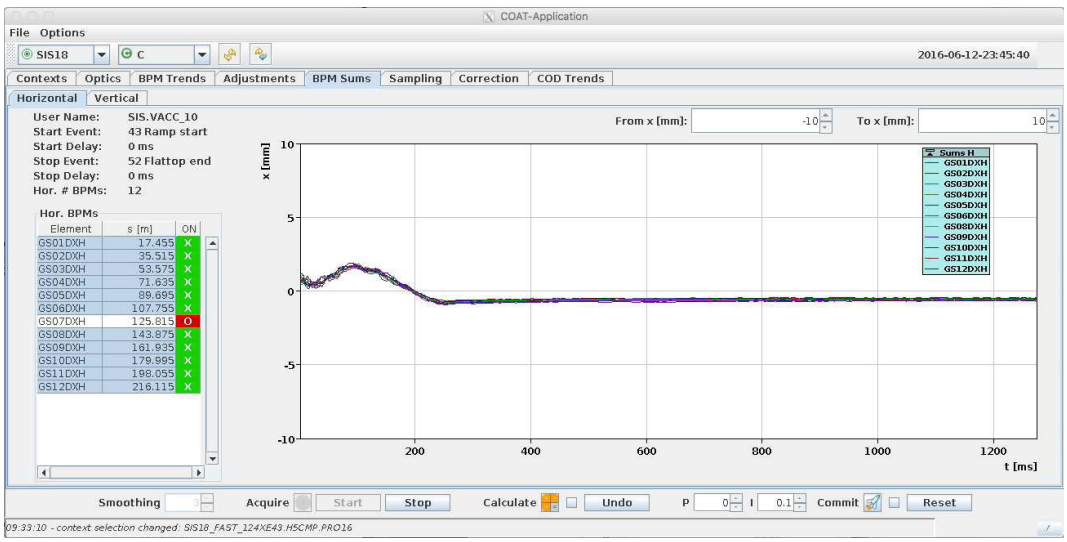


Figure 6: COAT Client, showing horizontal BPM sums signals after correction, while not accounting for dispersion.

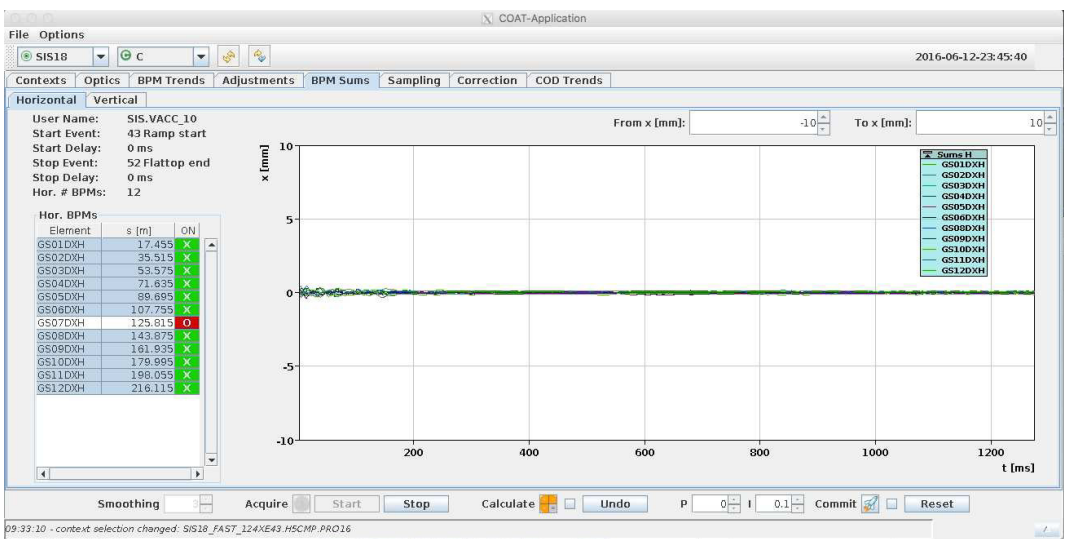


Figure 7: COAT Client, showing horizontal BPM sums signals after correction, with theoretical dispersion subtracted.