ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA137

# INTEGRATION OF A MODEL SERVER INTO THE CONTROL SYSTEM OF THE SYNCHROTRON LIGHT SOURCE DELTA

D. Schirmer*, A. Althaus†
Center for Synchrotron Radiation (DELTA), TU Dortmund University, Germany

## *Abstract*

During the past decades, a variety of particle optics programs have been applied for accelerator studies at the storage ring facility DELTA. Depending on the application, most programs were used offline without dynamic machine synchronisation. In order to centralize and standardize storage ring modeling capabilities, a dedicated online model server was developed and integrated into the EPICS-based control system. The core server is based on Python/EPICS service modules using OCELOT and COBEA as simulation tools. All data, actual machine readings/settings, conversion coefficients, results of simulation calculations as well as manual parameter settings, are handled via EPICS process variables. Thus, the data are transparently available in the entire control system for further processing or visualisation. To improve maintainability and adaptability, the remote presentation model controller concept was realized in the implementation. The paper explains the setup of the model server and discusses first use cases.

## INTRODUCTION

DELTA is a 1.5–GeV electron storage ring operated as a synchrotron light source by the TU Dortmund University [1]. Since 2011, a short-pulse facility for coherent subpicosecond pulses in the vacuum ultraviolet (VUV) and Terahertz regimes has been established [2, 3]. In recent years, various programs for storage ring modeling and tracking calculations were employed. Early accelerator design studies were carried out with the Pascal/Delphi program OPTICS [4]. Later, depending on the application focus, MAD [5] (optics and orbit correction studies [6]) and ELEGANT [7] (CHG/EEHG model calculations [8]) were used. These programs are usually applied in offline mode, i. e., without direct machine connectivity. For online modeling, the Accelerator Toolbox (AT) has proven to be useful [9, 10]. The AT is a collection of software tools to model storage rings and beam transport lines in a MATLAB workbench environment [11] and is mainly used as a stand alone program for prototypical applications. Direct read/write access to all EPICS [12] process variables (PVs), i. e., to all machine parameters, is enabled via the channel access (CA) protocol software interfaces 'labCA' [13] and 'mCA' [14]. In order to enable model calculations and storage ring simulations with live machine data, a novel Python/EPICS-based model server was integrated into the DELTA control system.

## MODEL SERVER SOFTWARE SETUP

The model server implementation makes use of the Remote Presentation Model (RPM) [15] software design pattern [16]. It is commonly applied for developing client-server-based user interfaces (UI). Like the Model View Controller (MVC) pattern [17], it divides software code into three interconnected parts (model, view, controller). Thus, it provides a more effective way to create clearly structured, more maintainable and reusable applications. The basic concept is shown in Fig. 1. The server side controller (e. g., optics calculator or conversion scripts, see next section) contains all the controller logic for a specific view (e. g., Python/Qt-based or Tcl/Tk-based GUI panels). The model data of this controller is automatically synchronized with the view data via the EPICS channel access (CA) protocol.

The RPM paradigm was largely used during the implementation of the DELTA model server. The data exchange between individual program modules as well as their configuration control is handled by dedicated EPICS PVs. Thus, all input/output simulation data as well as the software module configurations are transparently available in the entire EPICS control system.

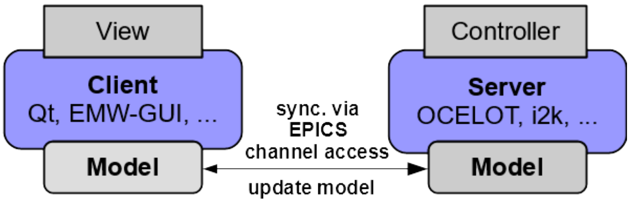


Figure 1: Remote Presentation Model (RPM) software design pattern used for the EPICS-based model server.

Since the EPICS-based DELTA control system is currently being converted from Tcl/Tk-based to Python-based applications [18], the simulation library OCELOT [19] was chosen as the modeling core software. OCELOT is a software framework for beam dynamics simulation studies mainly written in Python. The software package calculates, i. a., linear and nonlinear storage ring optics and requires a lattice description file. Therefore, a detailed lattice data file was created which models the DELTA storage ring as accurately as possible. It contains all currently (status 2019) installed storage ring components including drift spaces, dipoles, quadrupoles, sextupoles, slow and fast orbit steerer magnets, beam position monitors (BPMs), RF cavity, undulators as well as a dipole-like edge-model of the superconducting asymmetric wiggler magnet (SAW). This modeling input file was cross-checked by linear optics calculations with different simulation programs (Matlab-AT, MAD, OP-

* detlev.schirmer@tu-dortmund.de
† andre.althaus@tu-dortmund.de

TICS, ELEGANT). All programs compute identical Twiss parameters and radiation integrals (e. g., see Fig. 2).

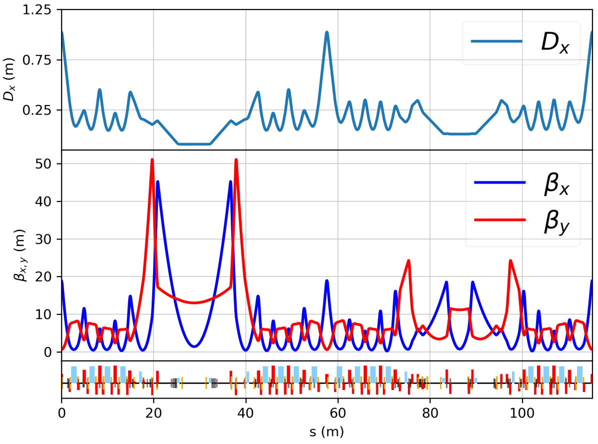


Figure 2: DELTA storage ring optics calculated by OCELOT [19] (design parameters, SAW off).

Furthermore, all existing auxiliary programming scripts for adjustment and conversion of machine data settings were rewritten in Python. An important tool is a script which converts magnet excitation currents into corresponding magnet model parameters (MPs), whereby each multipole component (i. e., magnet type) is assigned to dedicated model parameters. Several magnet types in the DELTA storage ring are assembled as combined function magnets, e. g., quadrupole magnets with additionally installed dipole correctors and integrated sextupole coils. The conversion script 'i2k' takes saturation and magnetic crosstalk effects due to the shared iron yoke into account. For more details see [6].

## SERVICE MODULES

Until now, the model server system consists of three EPICS service modules developed in Python, which function as independent softIOCs. Each softIOC can be initialized by individual configuration files (see Fig. 3):

- i2k-conversion-IOC: Determinates the magnet model parameters on the basis of magnet currents. The EPICS client side is realized with the module PyCA (Python Channel Access) [20]. 'PyCA' is a CA client module that provides a 'ca-pv' class to establish connections to a CA-server (i. e., epics-IOC). Thus it offers simple connections for Python applications with read/write access to EPICS PVs. This is used, for example, to read out the actual readback and set values of the magnet currents.
  EPICS PVs containing the model parameters are provided on the EPICS server side using the module 'PCASpy' [20–22]. This module offers not only the low-level Python connection to the EPICS CA server, but also a high-level abstraction to simplify the programming of server tools. Every change of magnet currents triggers the calculation of new model parameters (via EPICS monitors).
- OCELOT-simulation-IOC: The Python program package OCELOT calculates the storage ring optics. As input, the program needs a lattice file name defined in an EPICS PV. In addition, it reads actual machine status PVs as well as the actual model parameter PVs provided by the i2k-IOC via 'PyCA'.
  Two instances of simulation IOCs are implemented: online and manual. Both are initialized with different configuration files (online.conf and manual.conf). The simulation results are stored in EPICS PVs via 'PCASpy', an EPICS PV server (see above), e. g., arrays for the Twiss parameters, tune PVs, etc. Here again, every input PV change triggers a new OCELOT simulation.
- COBEA-IOC: Determination of optical functions using the program package COBEA. For more details see [23]. COBEA requires as inputs a unique drift space, the number and sequence of corrector magnets and BPMs as well as a measured response matrix whose file name is defined in an EPICS PV. The COBEA calculations are currently still static, since any model parameter variation requires an automated adaption of the response matrix, which is currently not yet implemented.
- IOC-calc: Specific magnetic field conversions which can not be handled by the i2k-IOC as well as other auxiliary EPICS records are provided by the dedicated softIOC 'IOC-calc' (i. e., implemented by EPICS calc-records).

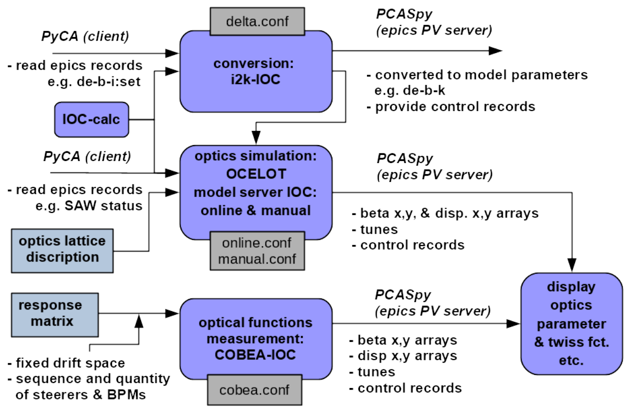


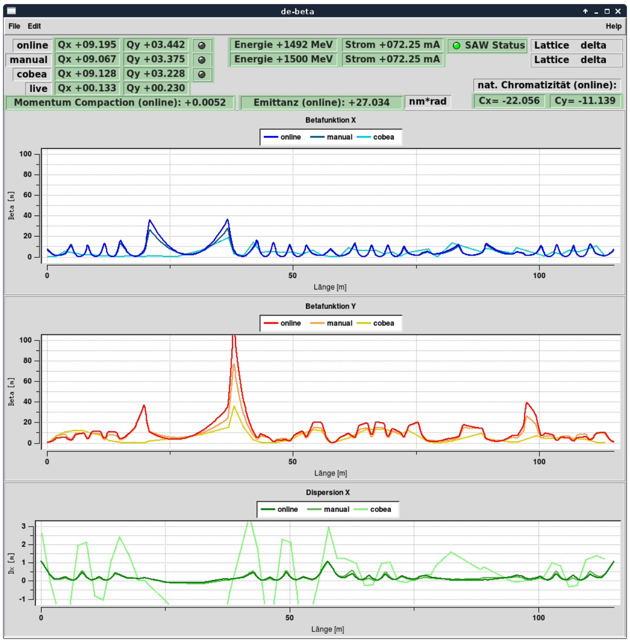
Figure 3: Setup of the Python/EPICS-based model server system.

The 'i2k'-module reads magnet currents, currently set during machine operation, converts them into MPs and writes them to EPICS PVs. They are continuously monitored and updated. On demand, predefined MPs in the OCELOT lattice file are individually replaced by online or manual (offline) values. For each online and/or offline variation, OCELOT starts a new model calculation. Here again, the simulation results are written to corresponding EPICS-PVs and thus are also available for further processing or visualization in the entire control system, like the i2k-converted model parameters mentioned above.

All programs and scripts are running as EPICS softIOCs in a Kernel-based Virtual Machine (KVM) [24]. The softIOCs processes are controlled by the ’systemd’ daemon, a background service manager in Linux operating systems [25]. Thus, any softIOC can be started independently. This feature allows the comparison of simulations with different input data, e. g., fixed design, manual (offline) or online values, as well as various lattice description or setup files. All results can be compared in a plot together with COBEA calculations (see next section).

## STORAGE RING MODELING

To test the functionality of the model server setup, at first only dipole and quadrupole magnets as well as the superconducting asymmetric wiggler (SAW) were considered. This enables studies on linear optics in different simulation modes as explained in last section. Figure 4 compares model server results for optical functions (beta functions, dispersion) and tunes in three modes: online, manual and COBEA. The online (i2k-based) simulation results for momentum compaction factor, emittance and chromaticity are displayed exemplarily. For more data a dedicated comparison table (i. e., GUI-panel) can be displayed. The calculations relate to the indicated machine parameters: energy (Energie), beam current (Strom) and SAW status (on/off). The underlying lattice file names are shown at the upper right (here: ’Lattice delta’ for online and manual mode).

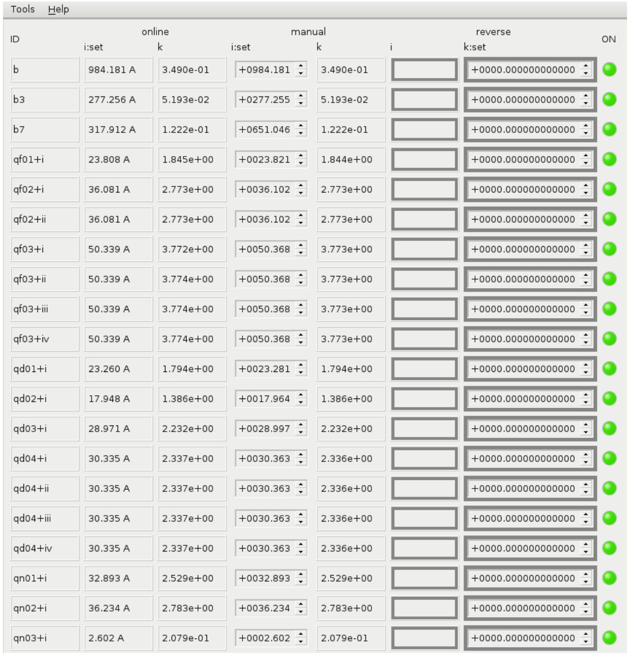


| | | | | | | | |
|---|---|---|---|---|---|---|---|
| online | Qx +09.195 | Qy +03.442 | ● | Energie +1492 MeV | Strom +072.25 mA | ● SAW Status | Lattice delta |
| manual | Qx +09.067 | Qy +03.375 | ● | Energie +1500 MeV | Strom +072.25 mA | | Lattice delta |
| cobea | Qx +09.128 | Qy +03.228 | ● | | | | |
| live | Qx +00.133 | Qy +00.230 | | | | | |

Momentum Compaction (online): +0.0052
Emittanz (online): +27.034 nm*rad
nat. Chromatizität (online): Cx= -22.056 Cy= -11.139

Figure 4: Comparison of optical-function calculations based on online (i2k-converted) and manual values (e. g., design parameter) as well as COBEA results. In addition, the corresponding tunes are displayed together with the live measurements of real machine data.

By default, the manual mode simulation uses predefined design model parameters specified in the associated lattice file. These data can be individually modified via the i2k-magnets GUI-panel (see Fig. 5). Each change in the panel triggers a new optics calculation. Simultaneously, the plots in Fig. 4 display the results for the online mode, i. e., for i2k-converted values. Here, magnet current changes at real power supplies trigger a recalculation of the optics. In addition, the static results of COBEA calculations are shown. It should be noted, that COBEA only outputs the beta functions at the 54 BPM positions. The reverse function (k2i), i. e., applying recalculated model parameters to the real power supplies of the magnets, is also possible.

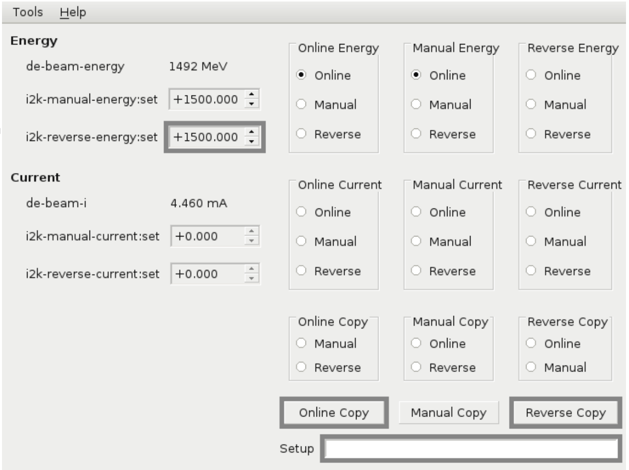


| ID | online | | manual | | reverse | | ON |
|---|---|---|---|---|---|---|---|
| | i:set | k | i:set | k | i | k:set | |
| b | 984.181 A | 3.490e-01 | +0984.181 | 3.490e-01 | | +0000.000000000000 | ● |
| b3 | 277.256 A | 5.193e-02 | +0277.255 | 5.193e-02 | | +0000.000000000000 | ● |
| b7 | 317.912 A | 1.222e-01 | +0651.046 | 1.222e-01 | | +0000.000000000000 | ● |
| qf01+i | 23.808 A | 1.845e+00 | +0023.821 | 1.844e+00 | | +0000.000000000000 | ● |
| qf02+i | 36.081 A | 2.773e+00 | +0036.102 | 2.773e+00 | | +0000.000000000000 | ● |
| qf02+ii | 36.081 A | 2.773e+00 | +0036.102 | 2.773e+00 | | +0000.000000000000 | ● |
| qf03+i | 50.339 A | 3.772e+00 | +0050.368 | 3.773e+00 | | +0000.000000000000 | ● |
| qf03+ii | 50.339 A | 3.774e+00 | +0050.368 | 3.773e+00 | | +0000.000000000000 | ● |
| qf03+iii | 50.339 A | 3.774e+00 | +0050.368 | 3.773e+00 | | +0000.000000000000 | ● |
| qf03+iv | 50.339 A | 3.774e+00 | +0050.368 | 3.773e+00 | | +0000.000000000000 | ● |
| qd01+i | 23.260 A | 1.794e+00 | +0023.281 | 1.794e+00 | | +0000.000000000000 | ● |
| qd02+i | 17.948 A | 1.386e+00 | +0017.964 | 1.386e+00 | | +0000.000000000000 | ● |
| qd03+i | 28.971 A | 2.232e+00 | +0028.997 | 2.232e+00 | | +0000.000000000000 | ● |
| qd04+i | 30.335 A | 2.337e+00 | +0030.363 | 2.336e+00 | | +0000.000000000000 | ● |
| qd04+ii | 30.335 A | 2.337e+00 | +0030.363 | 2.336e+00 | | +0000.000000000000 | ● |
| qd04+iii | 30.335 A | 2.337e+00 | +0030.363 | 2.336e+00 | | +0000.000000000000 | ● |
| qd04+iv | 30.335 A | 2.337e+00 | +0030.363 | 2.336e+00 | | +0000.000000000000 | ● |
| qn01+i | 32.893 A | 2.529e+00 | +0032.893 | 2.529e+00 | | +0000.000000000000 | ● |
| qn02+i | 36.234 A | 2.783e+00 | +0036.234 | 2.783e+00 | | +0000.000000000000 | ● |
| qn03+i | 2.602 A | 2.079e-01 | +0002.602 | 2.079e-01 | | +0000.000000000000 | ● |

Figure 5: Screenshot of the i2k-magnets GUI (dipole and quadrupole magnets only). The panel is used to manually set (i:set) individual magnet currents for offline simulations (see Fig. 4). The resulting model parameters (manual k) can be reconverted via ’k2i’ into new magnet current settings (reverse).

By means of the i2k-control GUI-panel (see Fig. 6) selected ’global’ machine parameters like beam energy or beam current can be changed individually and assigned to different simulation modes. An important feature is the possibility to exchange (copy) all model parameter settings between the different model server modes. Furthermore, stored setup files of real machine magnet settings can be read.

The GUI-panels ’i2k-controls’ and ’i2k-magnets’ are implemented using the recently developed Python-based display manager ’DeltaDM’. The optical functions are displayed by the standard Tcl/Tk-based Delta-EMW tool. Due to the EPICS-based RPM concept, all GUI-panels can easily be replaced by alternative GUI frontends.



**Energy**

de-beam-energy 1492 MeV

i2k-manual-energy:set +1500.000

i2k-reverse-energy:set +1500.000

**Current**

de-beam-i 4.460 mA

i2k-manual-current:set +0.000

i2k-reverse-current:set +0.000

Online Energy: ● Online ○ Manual ○ Reverse

Manual Energy: ● Online ○ Manual ○ Reverse

Reverse Energy: ○ Online ○ Manual ○ Reverse

Online Current: ○ Online ○ Manual ○ Reverse

Manual Current: ○ Online ○ Manual ○ Reverse

Reverse Current: ○ Online ○ Manual ○ Reverse

Online Copy: ○ Manual ○ Reverse

Manual Copy: ○ Online ○ Reverse

Reverse Copy: ○ Online ○ Manual

Setup

Figure 6: Screenshot of the i2k-control GUI. The panel is used, i. a., to set 'global' machine parameters (beam current, enery). For other notes see text.

To compare the different model calculation modes, the live measured tunes $Q_x, Q_y$ during accelerator operation are displayed (see Fig. 4, upper left). The calculated tunes are deduced from optical functions (integrated phase advance) and thus an appropriate quantity to justify different model simulations. Table 1 summarizes first results.

Depending on the applied response matrix, COBEA calculations are in good agreement with the measured tunes. Simulation results with the design model parameters (manual) and the online model are also satisfactory. Deviations are obtained in both modes, especially in the vertical plane (see Table 1).

Table 1: Comparison of Model Server Tune Calculations with Live Machine Data

| | **SAW off** | | **SAW on** | |
|---|---|---|---|---|
| | $Q_x$ | $Q_y$ | $Q_x$ | $Q_y$ |
| online | 9.125 | 3.358 | 9.111 | 3.392 |
| manual | 9.191 | 3.278 | 9.207 | 3.375 |
| COBEA | 9.128 | 3.228 | 9.128 | 3.228 |
| live | 9.130 | 3.233 | 9.133 | 3.235 |

Possible causes for these deviations are:

- Large orbit amplitudes in sextupole magnets result in tune shifts. Until now, real orbit distortions are not considered in the online accelerator model.
- Measurements have shown that some quadrupole power supplies are insufficiently calibrated (see [26]).
- The edge focusing of the superconducting asymmetric multipole wiggler causes a strong vertical tune shift. The magnetic field is modelled by a sequence of short rectangular dipoles, so far. This is only an approximation of the real asymmetric field characteristics.

Further detailed studies are needed on these issues.

## SUMMARY AND OUTLOOK

A first version of an Python/EPICS-based model server has been integrated into the DELTA control system. The implementation based on a modular and flexible client-server architecture according to the Remote Presentation Model software design concept. This allows, i. a., studies on storage ring models with various configurations in parallel. For example, online and offline simulations can be compared in real-time. In addition, magnet settings, obtained from simulation results, can be transferred to the real machine and vice versa. The data exchange is managed by EPICS PVs which are available transparently in the entire control system for further processing.

So far, only accelerator components have been considered for calculations on linear optics and chromaticity. Components for investigations on non-linear particle dynamics (e. g., sextupoles) and orbit correction (e. g., steerer magnets) are already integrated in the storage ring model and the model server provides corresponding conversions. Here, however, more work must be invested to calibrate the model parameters with respect to the machine hardware.

In the future, the DELTA model server could also serve as a basis to fit accelerator models to real machine data with the assistance of machine learning methods [27].

## ACKNOWLEDGEMENT

We would like to thank all colleagues of the DELTA team for many inspiring discussions, helpful suggestions and providing sufficient measuring time in accelerator shifts. Especially we would like to thank Stephan Kötter and Berhard Riemann for supplying the software library COBEA. Furthermore, we thank all colleagues who made the DELTA model server possible by providing a variety of software libraries, programming interfaces and documentations.

## REFERENCES

[1] T. Weis *et al.*, "Status of the 1.5 GeV Synchrotron Light Source DELTA and Related Accelerator Physics Activities", in *Proc. 20th Russian Particle Accelerator Conf. on Charged Particle Accelerators (RuPAC'06)*, Novosibirsk, Russia, Sep. 2006, paper MONO02, pp. 138–140.

[2] S. Khan *et al.*, "Coherent Harmonic Generation at DELTA: A New Facility for Ultrashort Pulses in the VUV and THz Regime", *Synchrotron Radiation News*, vol. 24, pp. 18, 2011.

[3] S. Khan *et al.*, "Generation of Ultrashort and Coherent Synchrotron Radiation Pulses at DELTA", *Synchrotron Radiation News*, vol. 26, pp. 25–29, 2013. doi:10.1080/08940886.2013.791213

[4] K. Wille, "Manual: OPTICS Version 4.00", Dortmund University, Germany, December 2004.

[5] H. Grote, F. C. Iselin, "The MAD Program (Methodical Accelerator Design)", Version 8.19, CERN, 1996.

[6] M. Grewe, "SVD-basierte Orbitkorrektur am Speicherring Delta", dissertation, Dortmund University, 2005.

[7] M. Borland, "elegant: A Flexible SDDS-Compliant Code for Accelerator Simulation", in *Proc. ICAP'2000*, Darmstadt, Germany, 2000.

[8] R. Molo, "Towards Echo-Enabled Harmonic Generation at FLASH1 and DELTA", dissertation, TU Dortmund University, 2017.

[9] ATcollab, http://atcollab.sourceforge.net/index.html.

[10] A. Terebilo, "Accelerator Modeling with Matlab Accelerator-Toolbox", in *Proc. PAC'01*, Chicago, USA, pp. 3203–3205.

[11] MATLAB/SIMULINK, Release 2017b, The MathWorks, Inc., Natick, Massachusetts, United States.

[12] EPICS, https://epics.anl.gov/index.php

[13] Using labCA, https://www.slac.stanford.edu/grp/ssrl/spear/epics/extensions/labca/manual/node30.html

[14] EPICS: mca, https://github.com/epics-modules/mca

[15] Remote Presentation Model (RPM), https://canoo.github.io/dolphin-platform/#_the_remote_presentation_model

[16] Software design pattern, https://en.wikipedia.org/wiki/Software_design_pattern

[17] Model View Controller (MVC), https://de.wikipedia.org/wiki/Model_View_Controller

[18] Python, https://www.python.org

[19] Ocelot, https://github.com/ocelot-collab/ocelot

[20] PyCA, https://github.com/slaclab/pyca

[21] PCASpy Documentation, https://pcaspy.readthedocs.io/en/latest/

[22] PCASpy Wiki, https://github.com/paulscherrerinstitute/pcaspy/wiki

[23] B. Riemann, S. Khan, S. Koetter, and T. Weis, "COBEA - Optical Parameters From Response Matrices without Knowledge of Magnet Strengths", in *Proc. IPAC'17*, Copenhagen, Denmark, May 2017, pp. 676–679. doi:10.18429/JACoW-IPAC2017-MOPIK066

[24] D. Schirmer, A. Althaus, P. Hartmann, and D. Rohde, "Control System Projects at the Electron Storage Ring DELTA", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 1361–1365. doi:10.18429/JACoW-ICALEPCS2017-THPHA013

[25] systemd, https://wiki.debian.org/systemd

[26] G. Dahlmann, H. P. Ruhl and G. Schmidt, "Vermessung der Ausgangsströme der DELTA Quadrupolnetzgeräte", Internal Report, DELTA, TU Dortmund University, 2019.

[27] D. Schirmer, "Orbit Correction With Machine Learning Techniques at the Synchrotron Light Source DELTA", presented at ICALEPCS'19, New York, USA, paper WEPHA138, this conference.