THE NTMAT EPICS-DDS VIRTUAL ACCELERATOR FOR THE CORNELL ERL INJECTOR

C. Gulliford, I. Bazarov, J. Dobbins, R. Talman, CLASSE, Ithaca, NY 14850, USA
N. Malitsky, BNL, Upton, NY 11973, USA

Abstract

Commissioning of the high brightness photoinjector for the Energy Recovery Linac at Cornell University continues [1]. To aid in this process we have developed a "Virtual Accelerator" application, which provides the beam physicist with an online high-level physics description of the machine. This application combines a linear optics model called Numerical Transfer Matrix (NTMAT), developed at Cornell, and EPICS-DDS, a middle-layer software based on the Experimental Physics and Industrial Control System (EPICS) toolkit and the Data Distribution Service (DDS) data-centric publish/subscribe model [2]. We present the initial results of implementing this new software tool and its deployment in the Cornell ERL injector control room.

THE NTMAT EPICS-DDS VIRTUAL ACCELERATOR APPLICATION

Before defining the term "virtual accelerator" it is instructive to first specify what we mean by an "online model." In our view, an online model of an accelerator consists of a simulation engine that reads in all relevant lattice settings from the accelerator control system and computes data corresponding to the real diagnostic data output from the accelerator. In this picture, there would ideally be a one-to-one map between all relevant control parameters and output data from the machine and its simulated counterpart. This model inherently assumes that the simulated lattice updates automatically any time the physical lattice settings are modified. This model represents an extremely valuable tool to the beam physicist by providing a useful way to benchmark the simulation engine as well as providing a detailed description of the dynamics in the accelerator in real time. However, it does not allow the beam physicist to directly use the simulation engine to guide beam operation because the virtual lattice settings are limited to those read in from the physical machine. It is this distinction that led us to identify the three key requirements used in the design and implementation of our application:

1. The ability to simultaneously and independently operate two accelerators in real time: one real and one virtual
2. The ability to synchronize the lattice settings in both machines
3. The ability to control and display relevant data from both machines in one integrated graphical user interface.

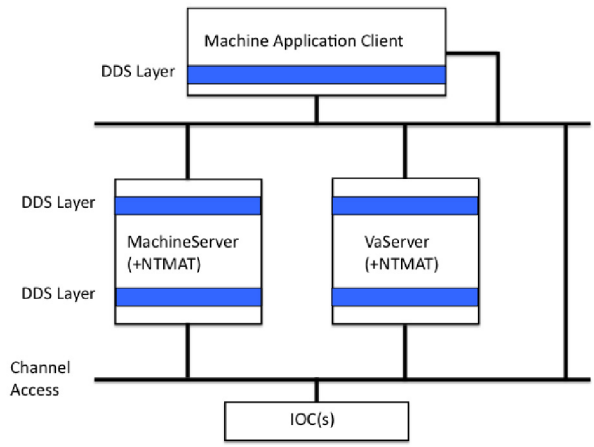

Figure 1: The virtual accelerator layout.

To fulfill these requirements we have followed a three-tiered approach common to many high-level accelerator control and simulation systems [2]. Figure 1 shows the basic layout of our implementation. The three tiers in this scheme include the distributed Input/Output Controllers (IOCs) serving data from the physical accelerator (bottom), the EPICS-DDS middle layer containing the NTMAT simulation engine (middle), and the Machine Client application (top). Each one of these components is described in detail in the following sections.

The Numerical Transfer Matrix Code

NTMAT is an object-oriented linear optics model and simulation library written in C++. The primary function of NTMAT is to compute the 6x6 transfer matrices through a user-defined accelerator lattice. In NTMAT the description of transport through optical elements falls into one of two general categories. The first is the standard 6x6 analytic transfer matrix description used in codes such as Transport [3, 4]. NTMAT currently supports the description of quadrupoles, dipoles, and vertical and horizontal corrector magnets in this way. The second method of element description is through field maps (FMAP elements). For a given FMAP element, the appropriate field map is read into the simulation and stored. The program then computes the first-order transverse gradients of the fields and uses them to calculate the infinitesimal transfer matrix $M(z \rightarrow z + dz)$. This technique also allows the fields from different elements to be superimposed, as is the case for several elements in the Cornell injector. NTMAT cur-

rently supports axial symmetric field maps for solenoids, standing wave RF cavities, and electrostatic elements. RF cavities with a two-fold mirror symmetry in the x-z and y-z planes are also supported. NTMAT can compute the 6x6 transfer matrices throughout a user-defined lattice, calculate the on-crest phases of the RF cavities, compute both the centroid orbit and the orbit of a particle offset from the centroid, and propagate the 6x6 RMS sigma matrix. For a more detailed description of the NTMAT simulation code, please see [5].

The EPICS-DDS Middle Layer

The current the control system for the Cornell ERL injector is EPICS [6]. In the EPICS control system, data are stored in IOC records called Process Variables (PVs). These records commonly store important information related to physical devices, such as a magnet power supply set points or read backs. Currently, EPICS stores this data in primitive datatypes such as integers, doubles, strings, as well as arrays of these primitive datatypes. Client applications can subscribe to these PVs, allowing them to publish/receive new values to/from the records. Similarly, in the DDS publish/subscribe model data providers publish typed data-flows identified by names called "topics," which data consumers can subscribe to in order to receive updated data as it becomes available. In EPICS-DDS, an extension of the EPICS toolkit, these topics become EPICS PVs, which can be published or served via Channel Access. This allows the same network protocol to be used between the distributed IOCs, middle-layer servers, and client applications. Additionally, EPICS-DDS allows users to serve complicated data structures based on the EPICS primitive datatypes. By providing the ability to send more complicated structures via Channel Access, the EPICS-DDS software represents a unified approach to serving data between various high-level object-oriented applications and a physical machine. For this reason we chose this software as the basis for our application.

The EPICS-DDS middle layer consists of two servers: the MachineServer and VaServer. The DDS topics used to communicate within the middle layer and to the Machine Client application include:

- *AccStrength*: an array of lattice parameter names and the corresponding element type-specific value (i.e. a quadrupole magnet's name and field gradient).
- *TBTData*: an array of x, y, and s positions representing a beam orbit.
- *MachineRequest*: a container specifying a request to the MachineServer application and a file name for data output.

The MachineServer application maintains and publishes the current state of the virtual lattice settings in the NTMAT beam line container. This server subscribes and publishes to the DDS topic "Machine:AccStrength," from which it receives updated virtual lattice settings from the user and publishes updated settings to the VaServer. The MachineServer application also subscribes to the "Machine:Request" topic. This allows the user to send specific requests to the MachineServer application, such as the "Save" request which dumps the current NTMAT lattice settings to a file. The second server in the middle layer is the VaServer. This application maintains and publishes the simulated orbit data based on the current NTMAT lattice settings received from MachineServer. Upon receiving the updated NTMAT strengths the VaServer calculates the new simulated orbit and publishes the results to the "VirtualAccelerator:TBTData" topic.

The Machine Client Application

The main graphical user interface for our virtual accelerator is the Machine Client application. This software is written in Java, using both the Java Swing toolkit and the JFreeChart library [7, 8], and consists of three separate applications: the Orbit Display window, the Strength Table window, and the Calibration Table window.

The main window of the Machine Client Application is the Orbit Display application, shown in Fig. 2. The graph in this window displays both the beam position readouts from the injector and the simulated orbit from the VaServer-NTMAT simulation. The actual data shown in Fig. 2 was taken during operation of the injector. To create this plot a bump in the simulated orbit was generated using several virtual horizontal corrector magnets. Next, the corresponding correctors in the injector were set to same values to try and recreate the bump in the BPM data. As seen in the graph, there is good agreement between the simulated/physical data except for the last three BPM read-outs. From this window, the user can save the NTMAT lattice and simulation settings in the MachineServer to a file, toggle between plotting the raw beam position monitor data or subtracting a saved reference orbit to form a difference orbit, or exit the Machine Client application (the Orbit Display application is the dispatching thread for all other windows).

The user can also start a Strength Table application, shown in Fig. 3, from the Orbit Display window. The table in the middle of the window displays the beam line element names, a corresponding brief description of the elements, the power supply type-specific settings in the NTMAT virtual lattice, the actual power supply settings of the physical elements in the injector, and the units of both values. Currently the values are displayed in the units of the physical power supplies, i.e. currents for the magnet elements in the injector. The virtual settings are converted to the NTMAT element-specific strength values using calibration data stored in the EPICS IOC. Both the values in the virtual and IOC setting columns can be edited in this table. Below the element data table are four buttons used to control both the virtual and physical element settings. The buttons in the top row are used to send the edited virtual/physical settings

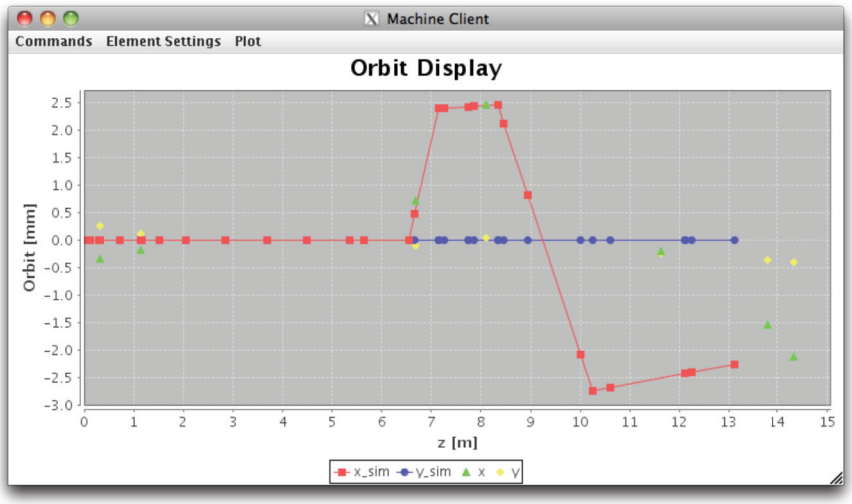

Figure 2: The orbit display application.

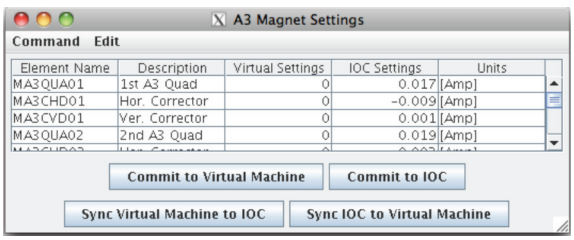

Figure 3: Strength table application.

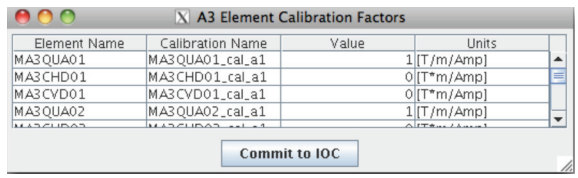

Figure 4: Calibration table application.

to the corresponding virtual/physical machine respectively. The buttons in the next row are used to synchronize the virtual lattice to the real lattice settings and vice versa. This combination of actions comprises the essential functionality of the virtual accelerator concept: the ability to independently "operate" two machines, the real injector and its high-level virtual counterpart. Users can also save the virtual machine settings from this window, retrieve the latest calibration data from the EPICS IOC, and start a Calibration Table application.

The Calibration Table window created displays the name, the corresponding calibration factor name, the value, and the units for the calibration factors of the lattice elements listed the parent Strength Table application. Figure 4 shows an example of a Calibration Table window. The user entered values in this table can be saved in the EPICS IOC. This ability to refine, edit, and save new calibration factors is critical in gaining a more complete understanding of the dynamics in the accelerator.

CONCLUDING REMARKS

We have created a new virtual accelerator application that allows users to simultaneously control both a physical and simulated accelerator. The application currently allows the user to display the orbit from the accelerator and the corresponding simulated orbit. This application has been installed in the Cornell ERL injector control room and initial tests of the software have been completed. Initial benchmarking of the software against orbit data from the injector shows some agreement. In addition to continuing this work, future plans for this project include the development of a matrix server application, similar in design to the VaServer, but with the ability to server matrix objects such as transfer matrices, response matrices, and the RMS sigma matrices.

REFERENCES

[1] F. Loehl et al., these Proceedings, presentation MOZRA01.

[2] J. Shah et al., "Prototype of a DDS-Based High-Level Accelerator Environment," ICALEPCS'09, October 2009, presentation TUA005, http://epics-dds.sourceforge.net.

[3] http://fermitools.fnal.gov/abstracts/transport/.

[4] A. Chao, M. Tigner, "Handbook of Accelerator Physics and Engineering," p. 73-75 (1999).

[5] C. Gulliford et al., "Linear Optics Modeling in the Cornell ERL Injector," PAC'09, May 2009.

[6] http://www.aps.anl.gov/epics/.

[7] http://java.sun.com/docs/books/tutorial/uiswing/.

[8] http://www.jfree.org/jfreechart.