*ORBIT DISPLAY'S USE OF THE PHYSICS APPLICATION FRAMEWORK FOR LCLS**

Michael Zelazny[#], Sergei Chevtsov[†], Chungming Paul Chu[‡], Diane Fairley[¤], Patrick Krejcik[∞], Partha Natampalli[Δ], Deborah Rogind[×], Greg White[☼], SLAC National Accelerator Laboratory, Menlo Park, CA, U.S.A.

Abstract

At the SLAC National Accelerator Laboratory (SLAC) the Controls Department (CD) is developing a physics application framework based on the Java(tm) programming language developed by Sun Microsystems. This paper will discuss the first application developed using this approach: a new Orbit Display. The software is being developed by several individuals in reusable Java packages. It relies on the Experimental Physics and Industrial Control System (EPICS) toolkit for data collection and XAL - A Java based Hierarchy for Application Programming for model parameters. The Orbit Display tracks and displays electron paths through the Linac Coherent Light Source (LCLS) in both a graphical, beam line plot, and tabular format. It contains many features that may be unique to SLAC and is meant to be used both in the control room and by individuals in their offices or at home. Unique features include BSA Beam Synchronous Acquisition (BSA), Orbit Fitting, and Buffered Acquisition.

TECHNOLOGIES

Reusable Java Packages

With an eye towards reusability, and the availability of several programmers, we adopted a divide and concur strategy. Thinking ahead to the various projects required of the Controls Physics-Applications group we partitioned the Java packages into several reusable components. Their dependencies are listed in Figure 1.

Being our first venture into the Java programming language, we learned, albeit the hard way, the two things Java programmers worry about the most:

1. The Java Classpath
2. The Java Threads

We discovered that certain things, such as handling the Graphics User Interface (GUI) button pushes and screen updates must be done in a special GUI thread, already present in your application courtesy of the Java programming language. The Orbit display's Java threads are described in Figure 2.

*Work supported by the U.S. Department of Energy under contract number DE-AC02-76SF00515.

[#]zelazny@slac.stanford.edu
[†]chevtsov@slac.stanford.edu
[‡]pchu@slac.stanford.edu
[¤]dfairley@slac.stanford.edu
[∞]pkr@slac.stanford.edu
[Δ]partha@slac.stanford.edu
[×]drogind@slac.stanford.edu
[☼]greg@slac.stanford.edu

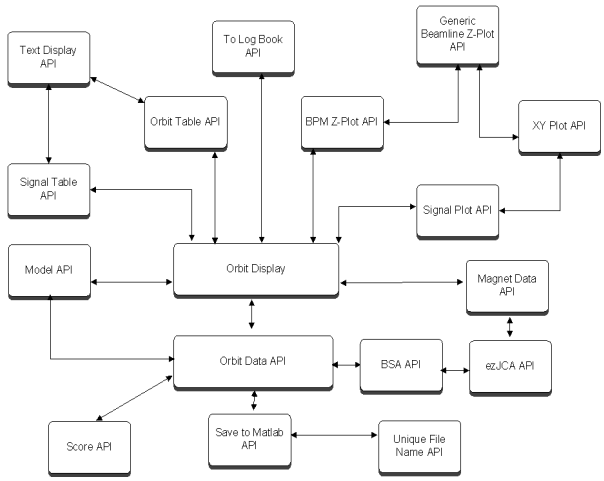

Figure 1: Java library dependencies.

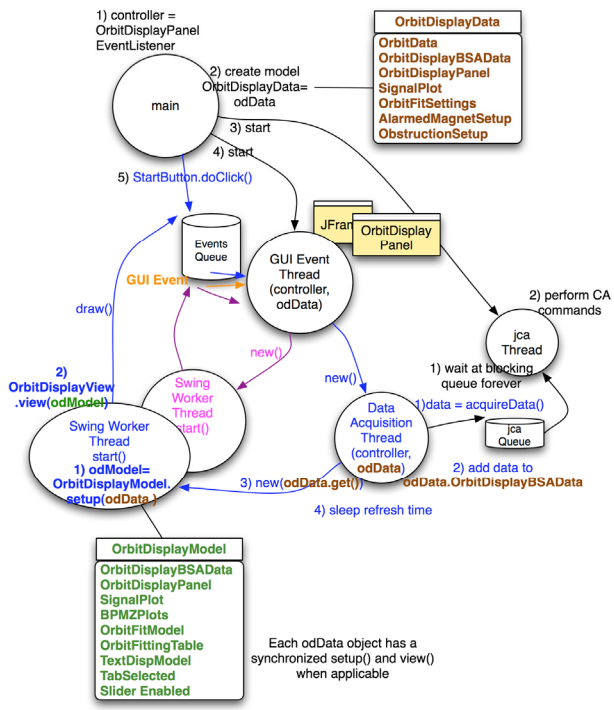

Figure 2: Java threads.

Eclipse

Due to its wide acceptance, and it low price, we chose Eclipse as our integrated development environment (IDE). Its build tool creates a file that contains a known working Java classpath. This a little shell script coaxing, this file can be used to properly set the CLASSPATH environment variable when launching the application.

EPICS

The LCLS has adopted the EPICS toolkit for new and upgraded beam line devices. To access EPICS process variables (PV) we chose to use Java Channel Access (JCA). JCA was chosen because it is a pure Java implementation of the EPICS CA protocol.

XAL

XAL, a high level accelerator application framework originally developed by the Spallation Neutron Source (SNS), Oak Ridge National Laboratory, provides generic hierarchical view for an accelerator [1]. XAL is also used to calculate the physics model used by the Orbit Display.

Standard GUI Framework

After considering frameworks as diverse as Eclipse and XAL, we decided that they lacked some of our desired features and were generally too complex to quickly build robust applications. So, we developed our own Graphical User Interface (GUI) Framework (GFW) [2]. In contrast, simplicity and customization are the main criteria behind GFW's existence. Using only basic Swing components, GFW defines a standard layout for GUI applications and allows the developers to extend the generic look-and-feel without any constraints (see Figure 3).

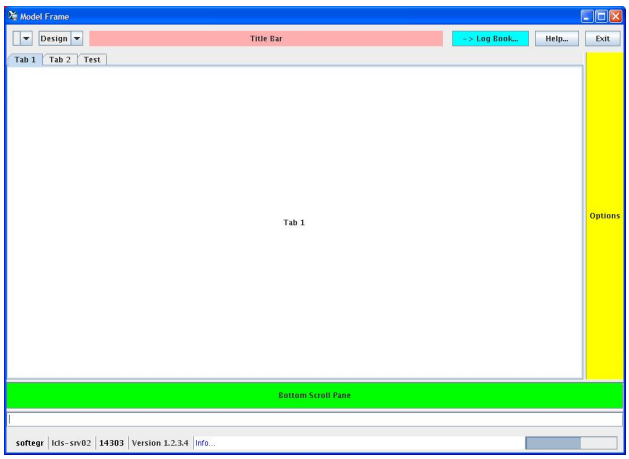

Figure 3: SLAC standard Java GUI framework.

STANDARD DISPLAYS

The Orbit Display application is the simplest extension of the standard beam line Z plot – devices displayed in the order the beam reaches them – Java package. This display shows a continuously updating portion of LCLS's beam position monitors (BPM), toriod charge monitors, out of tolerance magnets, beam line obstructions such as inserted profile monitors, and a cartoon representing the various beam line components, see Figure 4.

Since the standard beam line Z plot only gives the user a cursory overview of the various device's values, an updating table is also provided, see Figure 5.

Figure 4: Beam line Z plot.

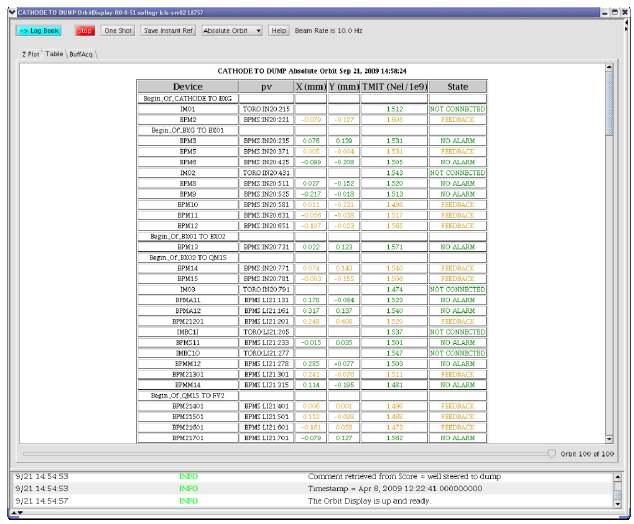

Figure 5: Orbit table.

FEATURES UNIQUE TO SLAC

Beam Synchronous Acquisition

SLAC physicists desire the ability to track a single particle bunch through the accelerator. Since the EPICS toolkit does not provide this functionality, we developed a facility to do this called Beam Synchronous Acquisition (BSA).

Orbit Fitting

Using a combination of measured parameters and the beam trajectory predicted by the accelerator model, it is possible to overlay the predicted beam path with the actual beam path, see Figure 6. Among other things, this is used to determine, for example, whether BPM signals are properly wired to the electronics used to read their signals.

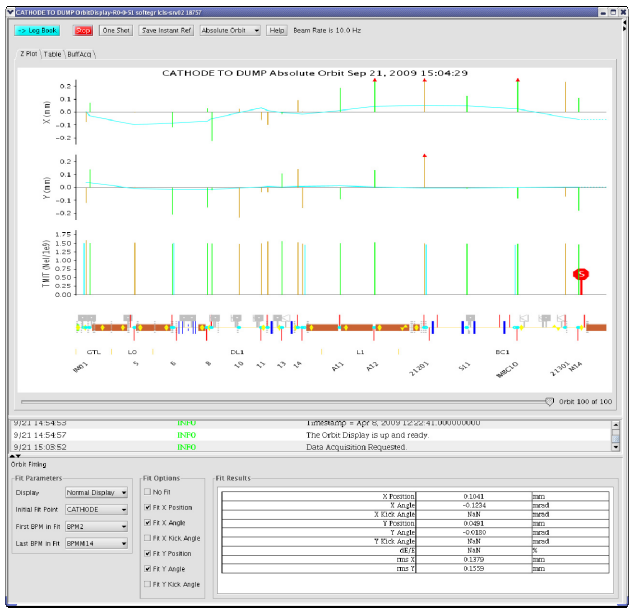

Figure 6: Orbit fitting.

Buffered Acquisition

In addition to viewing the beam trajectory through the accelerator, SLAC physicist's desire the ability to set up an experiment, run that experiment, then view how various signals changed. Since that LCLS machine rate is too fast to for our network to read every signal at the beam rate, the data is buffered on the EPICS Input Output Controller (IOC) then sent later to the applications requesting the data. Figure 7 is a plot of a BPM signal over time. Figure 8 is a table of the same signal. Figure 9 is a histogram of that signal.

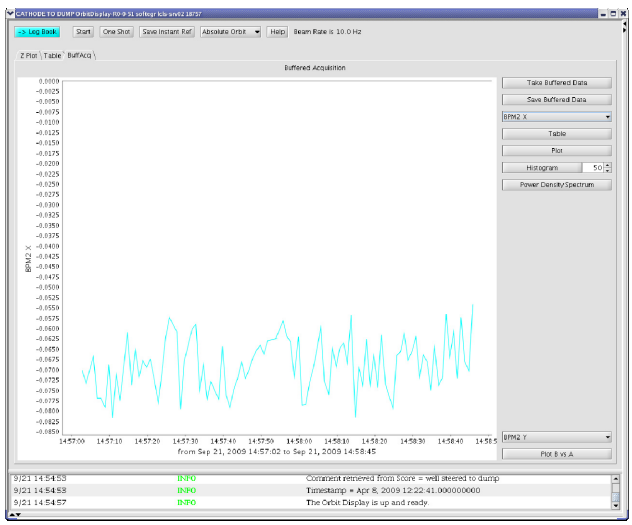

Figure 7: Signal plot.

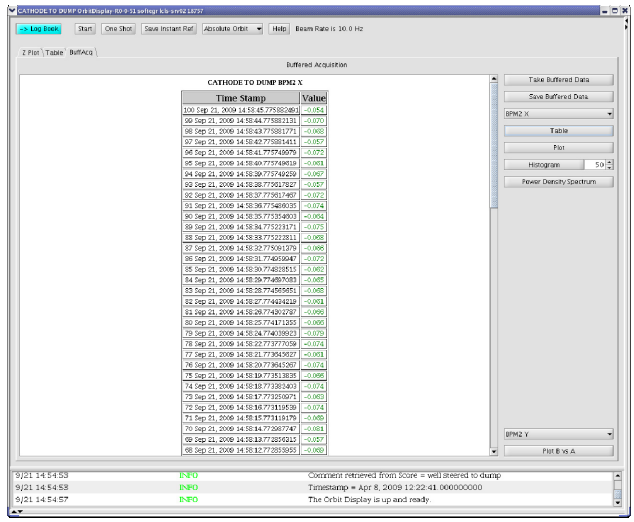

Figure 8: Signal table.

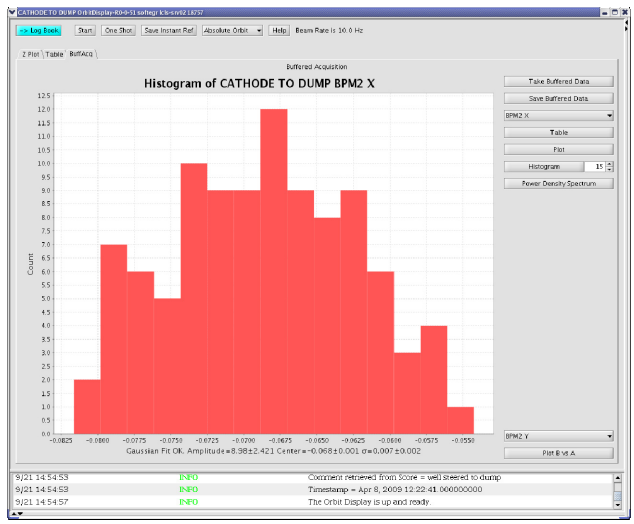

Figure 9: Signal histogram.

REFERENCES

[1] C. Paul Chu, "XAL Adoption Experience at LCLS", ICALEPCS '09, Kobe, Japan, October 2009, TUP012 http://icalepcs2009.spring8.or.jp/index.html.

[2] S. Chevtsov, "GFW - New GUI Framework at SLAC", ICALEPCS '09, Kobe, Japan, October 2009, THP103 http://icalepcs2009.spring8.or.jp/index.html.