XAL STATUS*

Thomas Pelaia II, Chris Allen, Sarah Cousineau, John Galambos, Jeff Holmes, Andrei Shishlo, Yan Zhang, Alexander Zhukov, Oak Ridge National Lab, Knoxville, TN 37830, U.S.A.
Chungming Paul Chu, Stanford Linear Accelerator, Menlo Park, CA 94025, U.S.A.

Abstract

XAL [1] is a Java framework for developing accelerator physics applications for the commissioning and operation of the Spallation Neutron Source (SNS) [2]. It was designed to be extensible and has evolved to support ongoing accelerator operations. In particular, the online model and applications have been extended to support the Ring. Core XAL design features eased the extension from Linac to Ring support and in some cases made it transparent. We discuss the recent advances and future directions in XAL and the current efforts to open the project to broader collaboration.

INTRODUCTION

XAL is an open source collection of Java packages developed for the commissioning and operation of the Spallation Neutron Source at Oak Ridge National Lab. Many developers from different facilities around the world have contributed code to the project. The project remains active and collaboration efforts are evolving as XAL adoption grows to include other laboratories. Currently, XAL includes over four dozen applications, numerous scripts, three services, an application framework, several tools and an online model for accelerator physics analysis. This paper will review some of the more recent and significant developments since the last published status report [3].

APPLICATION DEVELOPMENT

Background

The XAL application framework provides the foundation for applications which have a graphical user interface. It provides end users with applications that share a common look and feel while providing developers rapid development of document based applications. The developer provides three classes (application adaptor, document and main window) and three resource files (menu definition, help and application information) to participate in the framework. The three classes are subclasses that override their abstract parent base classes to provide hooks into the framework.

* ORNL/SNS is managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract DE- AC05-00OR22725

User Interface Design

Bricks is a new XAL application which assists the developer for rapid construction of user interfaces with compliance to the Model-View-Controller (MVC) design pattern. It draws from concepts used in OpenStep's™ Interface Builder™ [4]. Specifically, *Bricks* allows the developer to design views with the convenience of a graphical editor and it stores the designs in a XML file rather than generating source code. The developer can get a reference to any view within the design file. This is done in the controller class and it eliminates the need for writing source code for the views. In XAL, this means that the developer no longer needs to write the main window class for applications using the application framework.

Bricks has a code assistant that allows developers to paste view references right into their controller code independent of their Integrated Development Environment (IDE) of choice. *Bricks* even works well with Java based scripting languages such as Jython [5] and JRuby [6] since there is no code to compile for the user interface. Now, XAL developers can more easily design great looking user interfaces while focusing their efforts on application functionality.

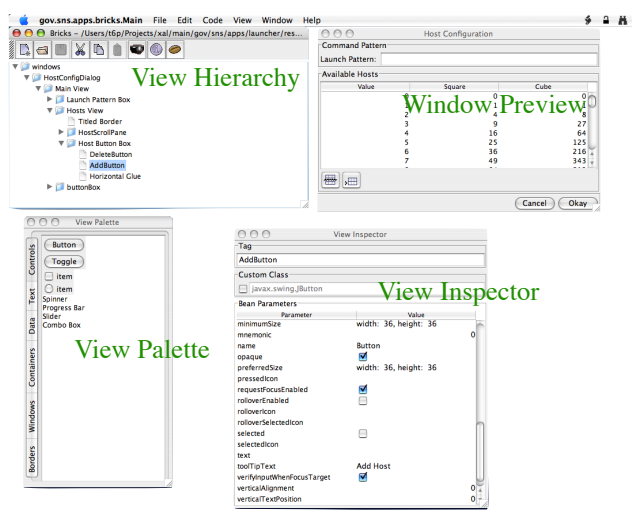

Figure 1: Bricks application.

Copy, Cut and Paste

From the beginning, the XAL application framework has supported automatic copy, cut and paste support for text through the associated *Edit* menu items which are provided by default. Automatic copy, cut and paste support has been extended to include any component that has transferable data and supports the standard Java transfer handler support protocol. When a component gains focus, the framework automatically handles enabling and disabling the copy, cut and paste menu items as appropriate based on the component's advertised ability to handle data transfer. When selected, these menu items initiate the associated data transfer events.

Improved Visual Cues

Many default menu item and tool bar buttons have been enhanced to use icons. Whenever available, standard Java icons are used. The use of icons improves the user experience and reduces the footprint of the toolbar buttons. The text that would have appeared in a button, now provides tool-tip text for that button.

Desktop Pane Support

XAL has always supported document based applications in which each document has an associated window which is free to be positioned anywhere on the screen. Some of our collaborators prefer to use desktop panes in which an application has a single containing window and each document has an associated internal window confined to the interior of that containing window. XAL now supports both kinds of applications. At SNS, the preferred window style is to have a single independent window per document.

APPLICATIONS

Several new applications have been developed, and we will discuss a few of them here along with some significant feature enhancements to existing applications. Besides the applications which are mentioned here, we also have numerous scripts many of which have graphical user interfaces.

Enhanced Applications

The *Wire Analysis* and *RTBT Wizard* applications have undergone significant enhancements.

The *Wire Analysis* application originally performed simple profile operations on wire scan data. It can now be used to performed more extensive beam analysis including matching using the solver and the online model to determine twiss parameters.

The *RTBT Wizard* provides analysis for several beam operations related to getting beam safely through the RTBT beam transport line and onto the mercury target. A new tab has been added that collects various target beam parameter measurements and allows the user to publish the parameters to the database. The user can also browse historical target beam parameters over a specified time range. Another tab has been added to provide an improved projection of the beam position on target based on multiple beam position monitors, the online model and historical data taken when we had a target view screen.

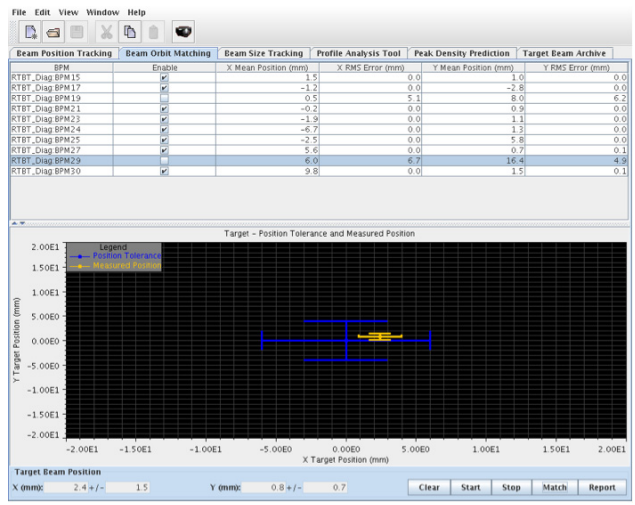

Figure 2: Target beam position projection.

Magnet Cycling

The *Magnet Cycling* application provides a convenient way to cycle bend magnets to remove hysteresis effects. For each magnet, the user can specify properties such as number of cycles and dwell time at the set points.

RF Simulator

The *RF Simulator* simulates both the RF controller and the beam in the ring. It mimics beam loading and the RF feed-forward and feedback processes to investigate beam stability and bunch leakage at high power.

Quad Shaker

One problem with our standard orbit correction application is that it defines the orbit as the measurements at the beam position monitors (BPMs) and it assumes that the BPMs are aligned to the quadrupoles. The *Quad Shaker* addresses these issues by determining the beam position relative to the quadrupoles and correcting the orbit by attempting to place the beam at the center of the quadrupoles.

Knobs Application

The *Knobs* application is a general application which allows the user to change multiple process variables concurrently and with specified coefficients relative to the knob value. Users can define multiple knobs within a

document and group the knobs within the document. Knob coefficients can be specified manually or through some automated procedures. A bump generator will generate bump knobs with the user specified number of elements per bump. The application also allows the user to generate coefficients automatically for a knob by taking before and after snapshots of the live process variable values corresponding to the knob elements. The resulting knob allows the user to easily change between two machine states.

Loss Viewer II

The application for viewing losses was rewritten from scratch. This new version is capable of supporting multiple types of loss monitors and currently supports both beam loss monitors and neutron detectors. Also, it can support multiple types of normalization. The new display allows the user to save the layout for multiple accelerator sequences in one document.

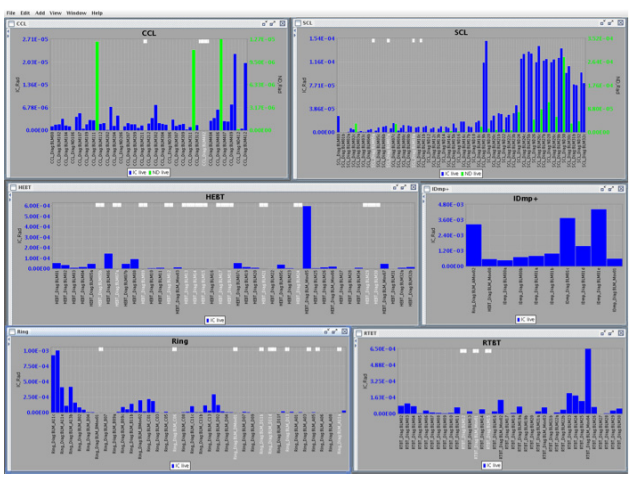

Figure 3: Loss Viewer II.

SERVICES

Services are distinguished from applications in that they do not have any graphical user interface and they run continuously on the server rather than on demand by the user. Applications and scripts can talk to services to get information and reconfigure them as desired.

Several significant enhancements have been made to the *PV Logger*. The logger is more robust by logging snapshots in memory and periodically publishing them to the database and retrying if the publishing fails. Individual applications can now request the service to log and publish a snapshot on demand. This eliminates the need for individual applications to run their own loggers.

The *Trip Monitor* is a new service that monitors process variables for trips and logs those trips to the central database. It uses an extensible set of rules to define what constitutes a trip. The *Trip Viewer* application is a new application which allows users to search for and view trips from the historical trip records and communicate directly with the trip monitor.

ONLINE MODEL

The online model in XAL is stable and typically gives good agreement with experimental analysis. What-If performance is dramatically improved due to the introduction of a cache for synchronized values. Bend magnets now have a reference angle so that the beam orbit distortion due to offset energy relative to design can be considered.

COLLABORATION

XAL has been a collaborative project from the beginning with roots in Brookhaven's Unified Accelerator Libraries (UAL) [7] and contributions and interest from labs around the world. However, the main effort of the project has focused on delivering applications for SNS. This has resulted in fragmentation of the source code among various laboratories. A recent effort intends to address this issue by explicitly creating an open source project to provide a common source base along with laboratory specific extensions. The Source Forge project named "xaldev" (http://sourceforge.net/projects/xaldev) provides the official forum for this effort, but much work remains to be done to create a viable candidate for common source code.

REFERENCES

[1] *XAL*, http://neutrons.ornl.gov/APGroup/appProg/xal/xal.htm

[2] ORNL Neutron Sciences, http://neutrons.ornl.gov/

[3] John Galambos, *et al.*, "XAL Application Programming Structure," http://www.jacow.org/p05/PAPERS/ROPA001.PDF, PAC 2005, Knoxville, TN

[4] Nancy Craighill, *OpenStep for Enterprises*, John Wiley & Sons, Inc., New York, NY, 1997.

[5] *The Jython Project*, http://www.jython.org

[6] *JRuby*, http://jruby.codehaus.org

[7] John Galambos *et al.*, "SNS Application Programming Environment," http://www.jacow.org/e02/PAPERS/THPLE021.pdf, Proceedings of EPAC 2002, Paris, France