ISBN: 978-3-95450-168-7 doi:10.18429/JACoW-IPAC2015-MOPWI050

OPEN XAL STATUS REPORT 2015*

T. Pelaia II#, C.K. Allen, A. Shishlo, A. Zhukov, ORNL, Oak Ridge, TN 37831, USA
Y.C. Chao, C. Gong, F. Jones, R. Newhouse, TRIUMF, Vancouver, BC V6T 2A3, Canada
P. Chu, D. Maxwell, Y. Zhang, MSU, East Lansing, MI 48824, USA
R. Fearn, L. Fernandez, E. Laface, M. Munoz, ESS, Lund, Sweden
J. Freed, University of South Carolina, Columbia, SC 29208, USA
P. Gillette, P. Laurent, G. Normand, GANIL, CAEN 14076, France
H. Hale, University of Tennessee, Knoxville, TN, USA
Y. Li, CSNS, Dongguan Campus, Institue of High Energy Physics, Chinese Academy of Sciences, Dongguan 523803, China
I. List, M. Pavleski, Cosylab, Ljubljana, Slovenia
P. Scruggs, East Tennessee State University, Johnson City, TN, USA

Abstract

Open XAL is an accelerator physics software platform developed in collaboration among several facilities around the world. The Open XAL collaboration was formed in 2010 to port, improve and extend the successful XAL platform used at the Spallation Neutron Source for use in the broader accelerator community and to establish it as the standard platform for accelerator physics software. The site-independent core is complete, active applications have been ported, and now we are in the process of verification and transitioning to using Open XAL in production. This paper will present the current status and a roadmap for this project.

INTRODUCTION

Open XAL [1] is an open source accelerator physics software platform written in Java. The Open XAL project began in mid 2010 as a response to requests from the international accelerator physics community to adopt an open source accelerator physics platform based on XAL [2] from the Spallation Neutron Source (SNS) at Oak Ridge National Lab (ORNL) and establish a standard platform for accelerator physics software.

The goals of the project are to provide a common accelerator physics core to be developed in collaboration, provide a rapid development environment and to modernize the source code. Since the last status report [3], the development effort has shifted from porting code to verification, modernization and new functionality. The API has stabilized, and the project is production ready. This paper covers the collective contributions of the collaboration.

SOFTWARE HIERARCHY

The software is grouped among the core, extensions, plugins, services, applications and scripts. The core consists of packages that include the accelerator object graph, the online model, abstract channel interface and general supporting tools such as common math libraries, archiving, messaging, database access and concurrency. This core is intended to be shared in common across all Open XAL distributions, and it has no compile time dependency on other components.

Extensions and plugins are components which may optionally depend on other extensions and plugins and the core. A plugin differs from an extension in that the core depends on a plugin being implemented (but not the actual implementation) whereas the core has no dependency on an extension. Plugins consist of controls channel access support and database adaptors. For example, the core specifies generic database tools, but a database adaptor plugin provides support for a specific database driver. Extensions include packages of general use in applications, scripts and services but not in the core. Examples of extensions include the application framework, service framework, lattice generation, fitting, scanning and widgets such as plotting tools.

Applications, scripts and services are executables and may depend upon the core, plugins and extensions. Applications and scripts are launched by users and have a user interface whereas services are headless and run perpetually in the background independent of user interaction. Applications differ from scripts in that applications are written in Java and are thus fully compiled prior to runtime versus scripts that may be written in a Java variant of Ruby or Python scripting languages and are not compiled prior to runtime.

A top level site directory contains optional site specific resources and configurations that take precedence over resources and configurations of the same name in the

* This manuscript has been authored by UT-Battelle, LLC, under Contract No. DE-AC0500OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for the United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).
# pelaiata@ornl.gov

ISBN: 978-3-95450-168-7 doi:10.18429/JACoW-IPAC2015-MOPWI050

main source tree and thus offer an opportunity for site specific customization.

COLLABORATION

The active collaboration consists of the China Spallation Neutron Source (CSNS) in Dongguan, China, European Spallation Source (ESS) in Lund, Sweden, Facility for Rare Ion Beams (FRIB) at Michigan State University in Lansing, MI, Grand Accélérateur National d'Ions Lourds (GANIL) in Caen, France, SNS in Oak Ridge, TN, and TRIUMF in Vancouver, Canada. Members from each facility contribute code, report issues and participate in workshops and monthly online meetings.

Code Sharing

The code is placed in Git [4] repositories [5] hosted on SourceForge and a ticket system is used to track tasks. The Git subtree feature is used extensively so subtrees of the code can be shared while allowing each site to choose which applications, scripts, extensions, plugins and services to include in their site specific branch.

SITE STATUSES

The operational status varies across the facilities and hence each facility has different needs for Open XAL, and the status at each site varies accordingly.

CSNS

MEBT commissioning for CSNS will use applications based on a mature XAL variant rather than Open XAL due to the confluence of commissioning with the nascent development of Open XAL. Several new applications have been developed to accommodate the Rapid Cycling Synchrotron (RCS), magnet database, lattice database, etc. These applications will be ported to Open XAL in the near future.

ESS

The Beam Physics group at ESS decided in November 2014 to use Open XAL for beam commissioning. The Open XAL model has been modified to handle field maps for cavities [6] and was benchmarked against TraceWin [7]. A python integration environment based on JPype [8] has also been developed. The Open XAL solver extension was used for matching the initial parameters of the beam based on ESS specific criteria. Automated installation scripts were created for Mac and Linux operating systems. Optics files are generated as custom XML files called "LinacLego" and are convertible to standard Open XAL format using an extension.

FRIB

Open XAL is currently a backup option for high level accelerator physics applications at FRIB, and thus has an uncertain future here. Several Open XAL applications have been tested in the control room during the September 2014 commissioning of a new cryomodule successfully demonstrating use of Open XAL for linac commissioning and beam tuning. A model platform was developed which allows integration with IMPACT [9] and MADX [10] as additional options for modeling.

GANIL (SPIRAL2 Project)

SPIRAL2 has adopted the OpenXAL September 2014 snapshot. Hibernate is used for database interaction, and the accelerator input XML files are generated from the database. The core has been modified slightly to support the concept of equipment in the accelerator object graph. Tools were developed to allow user interaction with process variables. Custom SPIRAL2 application adaptor and document subclasses have been developed as foundations for applications adding support for such things as displaying the accelerated beam label in the title bar, automatic combo-sequence selection, online/offline accelerator selection and the Nimbus look and feel. Several applications have been developed and others are being developed.

SNS

At SNS, the old XAL code has been frozen with the exception of critical bug fixes if necessary. All active applications (several dozen) have been ported to Open XAL, and it is now deployed as the default high level accelerator physics software platform. Several applications have been debugged and verified as production ready. XAL is still available as a fallback should critical bugs be encountered.

TRIUMF

At TRIUMF, a low energy empirical model has been successfully tested, and a magnet dithering and beam based alignment tool have been developed. The Experiment Automator application has been developed for automated data acquisition and contributed to the Open XAL community. An orbit correction package has been planned.

DEVELOPMENT HIGHLIGHTS

Several of the many developments since the last status report will be summarized here. Specifically, the ones discussed here are of common use across the sites.

Continuous Integration

ESS has setup a Jenkins continuous integration server for Open XAL providing analysis which has been used to improve the code.

Online Model

The online model has undergone major architectural improvements [11] which we briefly highlight here. Processing of simulation data has been separated from probe state population. This approach keeps analysis data separate from the trajectory and allows for richer simulation computations on the trajectory and its probe states.

Probes, probe states and trajectories have been modified to use Java Generics [12] which allows for compile time type checking and consistent typing across these classes.

ISBN: 978-3-95450-168-7 doi:10.18429/JACoW-IPAC2015-MOPWI050

The performance of fetching states from a trajectory has been improved significantly, and is especially noted when working with trajectories over a large number of states.

The lattice generator which maps model elements to device nodes has been rewritten making it simpler to maintain the code and support new device types.

The elapsed time mechanism has been refactored for RF Gaps leading to more sustainable code and removing the use of static variables to maintain state.

Site Customization

Site specific resources can be added and override existing resources of the same name if any. Site specific build properties can be specified and take precedence over the defaults.

Application Framework

The application framework now supports applications written entirely in a Java based scripting language such as JRuby [13] or Jython [14].

Services Framework

The services framework [15] now implements JSON-RPC [16] over the WebSocket [17] protocol which allows for client side web applications to natively communicate with Open XAL services.

Tools

One old external library of open source widgets that had been included in XAL contained a very useful wheel switch which was the only control in use from that library. The code for the wheel switch was modernized, issues addressed and brought directly into the Open XAL source thus eliminating dependence on that external library.

Applications

Significant improvements have been made to the Virtual Accelerator application including, adding solenoid support, allowing set points to be edited directly and adding a data plot for beam size and positions. A new application that allows for experimental automation was developed and contributed to the master branch.

FUTURE PLANS

Here are a highlights of a few of the many tasks that lay ahead for Open XAL. Currently Open XAL depends on Java 7. Soon, we plan to embrace Java 8 as our new dependency so as to allow for new Java 8 constructs and packages in the code base.

Online Model

Plans are for each RF Gap to be able to maintain its own transit time factor functions. Also, the model needs to be benchmarked against other established machine codes.

JavaFX

With the adoption of Java 8, we can embrace JavaFX [18] as a natural replacement for the Swing-based Bricks [19] GUI builder and framework. Support for JavaFX will also be added to the application framework.

Unit Tests

Support for unit tests will be extended to other components beyond just the core.

ACKNOWLEDGMENT

We are grateful for the leadership contributions of Sheng Peng of FRIB who organized online meetings and drove early momentum of the Open XAL collaboration.

REFERENCES

[1] Open XAL website: http://xaldev.sourceforge.net

[2] J. Galambos et al., "XAL Application Programming Structure," Proceedings of PAC 2005, Knoxville, TN (2005).

[3] T. Pelaia II, "Open XAL Status Report 2013," MOPWO086, these proceedings, IPAC'13, Shanghai, China (2013).

[4] Git website: http://www.git-scm.com

[5] Open XAL Git Repositories website: http://sourceforge.net/p/xaldev/_list/git

[6] E. Laface, I. List, "Field map model for the ESS Linac simulator," MOPJE031, these proceedings, IPAC'15, Richmond, VA (2015).

[7] TraceWin website: http://irfu.cea.fr/Sacm/logiciels/index3.php

[8] JPype website: http://jpype.sourceforge.net

[9] IMPACT website: http://amac.lbl.gov/~jiqiang/IMPACT/index.html

[10] MADX website: http://madx.web.cern.ch

[11] C.K. Allen et al., "Architectural Improvements and New Processing Tools for the Open XAL Online Model," MOPWI047, these proceedings, IPAC'15, Richmond, VA (2015).

[12] G. Bracha, "Generics in the Java Programming Language," July 2004; http://www.oracle.com/technetwork/java/javase/generics-tutorial-159168.pdf

[13] JRuby website: http://jruby.org

[14] Jython website: http://www.jython.org

[15] T. Pelaia, "Open XAL Services Architecture," MOPWI049, these proceedings, IPAC'15, Richmond, VA (2015).

[16] JSON-RPC website: http://json-rpc.org

[17] WebSocket Protocol website: http://tools.ietf.org/html/rfc6455

[18] JavaFX website: https://www.oracle.com/technetwork/java/javase/overview/javafx-overview-2158620.html

[19] T. Pelaia II, "XAL Application Framework and Bricks GUI Builder," Proceedings of ICALEPCS 2007, Knoxville, TN (2007).