APPLICABILITY OF XAL FOR ESS

Jaka Bobnar, Cosylab, Ljubljana, Slovenia
Steve Peggs and Charles Garrett Trahern, ESS, Lund, Sweden
Todd Satogata, Jefferson Lab, Newport News, Virginia, U.S.A.
Thomas Pelaia II and Christopher K. Allen, ORNL, Oak Ridge, Tennessee, U.S.A.

Abstract

XAL is a Java-based application framework, developed at the Spallation Neutron Source (SNS). The framework is designed to provide an accelerator physics programming interface to the accelerator, and it allows creation of general-purpose applications dedicated to various parts of the accelerator.

The backbone of the XAL framework is an XML-based description of the accelerator. The XML file provides the list of all devices, their properties, and relationships between devices within the system. Since the accelerator structure is defined in the relational database, XML can be generated directly from the database using appropriate adapters. This allows the framework to be more generic and enables it to run on different sites using various configurations.

The generality of XAL and the rich set of applications and tools provided by SNS make the framework very appealing for use at other accelerator sites. The European Spallation Source (ESS) is being built in Sweden, and is similar in complexity to the SNS. XAL has therefore been considered for use at ESS for high-level applications. The applicability of XAL and prototyping for ESS are discussed in this article.

INTRODUCTION

The XAL framework was developed by SNS as a part of their accelerator physics activities. It was designed to provide a common set of tools and applications used in machine physics and accelerator control. Today the framework includes a vast set of applications such as Orbit Correction, Wire Scanner Analysis, Scanning Application etc. These applications are all used in day-to-day activities in the SNS control room.

XAL was designed from the start to be as independent from machine details as possible. Therefore a specific model was defined which provides a detailed description of the accelerator. At start-up the model is parsed and used by the framework to gain access to various parts of the accelerator. The model allows XAL use at different accelerator sites without changing the code, since the model is supplied as a set of configuration files and is the only part of the framework that needs to be adapted.

Recently XAL went under major restructuring in order to make the code even more transparent and to allow easier development of site specific applications and components. ESS, being a similar machine to SNS, appeared as a potential heavy user of this framework (now named Open XAL).

ACCELERATOR MODEL

The backbone of the XAL framework is the accelerator model. The model describes the layout of the accelerator and its parameters as they are used by the applications.

The XAL model is defined in a hierarchical structure within an XML file. This XML file is composed of several different accelerator sequences, which consist of other sequences or components each describing a particular segment of the accelerator. Combined together, they form a hierarchy of the complete accelerator down to every particular device that can affect the beam path. In addition, the XML file also provides all the necessary pieces of information required for the control of a particular physical device. For example, the magnet description includes the strength of the magnetic field, its position within the accelerator, the power supply associated with it etc. [1].

XAL uses EPICS as the underlying control system to communicate with the accelerator hardware. EPICS communication uses a single “Process Variable” (PV) as the fundamental unit for communication with high-level software via a protocol called Channel Access. Therefore, in addition to the physical description of the devices, the XML model also carries information about associations between EPICS PVs and accelerator devices. A single device can have several different PVs, each assigned to one particular device attribute.

Based on the information in the XML file, a Java model is constructed by the XAL upon start-up of an application. Each component within the XML structure is mapped to a Java device object and can be treated as such in XAL applications. Users can set or read the attributes associated with any of those devices simply by changing the value of a particular field in that object, and changes are immediately reflected in the real system through the PV registered for that particular attribute.

Though XAL currently supports only EPICS control system, the underlying mechanism is abstracted so Channel Access can be replaced by other communication protocols. This permits some aspects of XAL to be truly portable between accelerator sites.

Taking into account all the aforementioned pieces of information, one can end up with an enormous XML model, which might be very difficult to maintain. Therefore, XAL works together with the central database, which stores all the required information. A dedicated XAL application gathers the information from the database and generates the XML file. This ensures that the model is always consistent with the accelerator and

makes it very easy to implement the changes that were introduced to the physical machine, if those are registered in the database. The structure and type of the database are not prescribed, since an XML generator can be written specifically for each system. This increases portability by only requiring site-specific adaptors to extract database contents to XML. In addition, the same database can also be configured to be used as a save/restore point for control system variables allowing users to save current state of the machine and restore it at any time later or any other feature, which might need to store or load information.

ONLINE MODEL

A critical component of many beam commissioning activities involves comparison of measured quantities with model predicted values. To facilitate this, XAL includes an "online" model, which is a simple envelope tracker designed for use in applications. This model implements on-the-fly calculation of beam parameters based on the machine settings [2-4].

The online model is loosely coupled with other parts of the XAL framework. The main components are the lattice (constructed from the aforementioned accelerator model) and a probe (describing the beam and how it is to be modeled). The lattice is generated via a set of rules from the accelerator node device information (generated from the XML model) and probe information is supplied via another configuration file.

Based on this information one can use XAL to perform simple simulations of the beam behavior inside the accelerator and tweak simulated machine parameters to achieve a desired response, then use the same framework to send the desired settings to the accelerator. The online model can also be used outside of XAL, as long as the lattice and probe information are provided.

APPLICATION FRAMEWORK

The XAL Application framework is a framework for rapid development of applications with a common look and feel, which provides many features that users expect from modern applications [5].

The application framework provides a set of classes that the applications extend to use common XAL features. The framework is based on Java Swing GUI components, and provides a simple GUI builder called *Bricks*, which can be used to build XAL applications even by developers who are not experienced Java programmers.

Through the framework, developers can access various common parts such as the accelerator model, the online model etc. Putting it simply, the framework provides a complete user interface to the accelerator.

Using the XAL framework has several advantages. The most important is a consistent look and feel of all applications used by the operators and therefore, minimization of the troubles that could appear if each application had slightly different layout, menu orders, toolbars etc. Furthermore, many features (such as copy, cut & paste, printing etc.) are automatically provided, easing the load on the developer. Those features can simply be turned on or off and the developer can focus more on the application content and less on implementation details.

In addition, the Open XAL project will support localization. Each application will provide an externalized text file where all the text (menus, buttons, labels etc.) will be located. By replacing the file with a translated one, users will be able to tailor the applications to their needs. This will contribute to easier use of XAL at sites where English is not the primary language.

EUROPEAN SPALLATION SOURCE

The European Spallation Neutron Source (ESS) is a project to design and construct a next-generation facility for research with beams of neutrons. At 5 MW beam power, The ESS will be the brightest source of neutrons in the world, enable scientists across many disciplines to perform experiments and investigate materials. The ESS will also retain and strengthen the current European position in the neutron science [6].

The ESS will be composed of a high-current proton linac, which will deliver 5 MW of power to the target at 2.5 GeV, with a nominal current of 50 mA [7]. With respect to the controls conceptual design, the machine will be similar to SNS. ESS has therefore planned to take advantage of experience and expertise developed at SNS, including standardization of a Controls Box environment for distributed R&D and development among partner laboratories [8], and use of the XAL framework as a solution for high-level applications and accelerator physics tool development.

XAL AT ESS

An XAL Workshop was held at SNS in May 2010, where current and potential future users of XAL discussed future goals and framework development. Attendees represented 11 organizations, including ESS, SNS, FRIB, BNL, TRIUMF, and CSNS [9]. Part of the outcome of the workshop was the previously mentioned plan for refactorization of the XAL libraries and structures. This refactorization should encourage development by users other than SNS. It will introduce much more modularity and easier maintenance; cooperation and sharing among users will also be easier.

Open XAL will be split into several different parts, each responsible for a particular group of functionalities (e.g. separating the core from the devices model, fully detaching the database access layer, etc.). Each user of XAL could then decide what modules to include in their distribution, which devices implementations are required by the machine, and so on.

At present, ESS has a partially constructed lattice database. Multi-particle beam dynamics for the linac has been studied using the TraceWin code [10]. The results of these studies and simulations have been entered into the MySQL database. A dedicated Java application has been

written to gather the data from the lattice database and constructs the accelerator model from the Java objects. This model can be serialized to an XML file, using the document definition specified by XAL. All the devices and other data that are required by XAL (e.g.. power supplies for magnets, epics channel names), but are yet missing in the lattice database are filled with dummy data to enable XAL model use at the earliest development stages.

Because there is no available ESS EPICS database yet, the model can only be used in simulation mode. A dedicated application called *Virtual Accelerator* is provided by XAL, which loads a specific accelerator sequence and simulates EPICS channels using the Channel Access Server (CAS) [11]. The values simulated by the CAS are calculated by the online model. Due to the nature of the CAS, the simulated channels can be used as any other EPICS PV and therefore, XAL can also connect and use those channels directly without the need to modify any part of the code.

The virtual accelerator feature will play an important role during the development and commissioning period of the ESS accelerator. It permits tests of features and the design of the accelerator without the need to connect to a fully implemented control real system. Users will be able to develop software without concerns about early integration problems.

The next step in adaptation of XAL for ESS is the addition of new devices and potential adaptation of the existing devices. ESS will use certain types of physical devices that are not used at SNS and are therefore non-existent in XAL. These new devices will have to be implemented and added to the XAL model. The optical properties of the new devices will also have to be implemented to allow use of the devices in the online model to perform beam dynamics simulations.

CONCLUSIONS

XAL has been a collaborative project from the beginning with roots in Brookhaven's Unified Accelerator Libraries, with contributions and interest from other labs around the world [12], though the main effort of the project has been to deliver applications for SNS. This has resulted in fragmentation of the code among various contributors. There has been a recent effort to organize and coordinate the project. XAL's position as a useful accelerator application framework will be strengthened by new laboratories joining the collaboration. The use of a well-developed framework and tested applications will improve early adoption of control application standards, and should ease the commissioning period of the ESS.

REFERENCES

[1] J. Galambos et al, "XAL Application Programming Framework", ICALEPCS 2003 Proceedings
[2] J. Galambos et al, "XAL Application Programming Structure", PAC 2005 Proceedings
[3] J. Galambos et al, "XAL – The SNS Application Programming Infrastructure", EPAC 2004 Proceedings
[4] C. M. Chu et al, "SNS Application Programming Infrastructure and Physics Applications", APAC 2004 Proceedings
[5] T. Pelaia II, "XAL Application Framework and Brick GUI Builder", ICALEPCS 2007 Proceedings
[6] http://www.ess-scandinavia.eu/
[7] M. Eshraqi et al, "Conceptual Design of the ESS LINAC", IPAC 2001 Proceedings
[8] T. Satogata et al, "ESS Controls Strategy and the Control Box Concept", these proceedings.
[9] http://neutrons.ornl.gov/conf/XAL2010
[10] http://irfu.cea.fr/Sacm/en/logiciels/index3.php
[11] http://caj.cosylab.com
[12] T. Pelaia II et al, "XAL Status", ICALEPCS 2007 Proceedings