SNS APPLICATION PROGRAMMING INFRASTRUCTURE AND PHYSICS APPLICATIONS*

C.M. Chu, S. Cousineau, J. Galambos, T. Pelaia, A. Shishlo, ORNL, Oak Ridge, TN, USA
C.K. Allen, C. McChesney, LANL, Los Alamos, NM, USA
D. Ottavio, BNL, Upton, NY, USA
W.-D. Klotz, ESRF, Genoble, France
I. Krisnar, A. Zupanc, Cosylab, Ljubljana, Slovenia

Abstract

The Spallation Neutron Source (SNS) is using a Java based hierarchal framework for application program development. The framework, called XAL, is designed to provide an accelerator physics programming interface to the accelerator. Much of the underlying interface to the EPICS control system is hidden from the user. Use of this framework allows writing of general-purpose applications that can be applied to various parts of the accelerator. Also, since the accelerator structure is initiated from a database, introduction of new beam-line devices or signal modifications are immediately available for all XAL applications. An online model is included in this framework for quick beam tracking. Simple interfaces to other external modeling software are also available. Standard graphical user interface (GUI) layout is provided for common look-and-feel purpose. Direct scripting interfaces are available for both Jython and Matlab™, for rapid prototyping uses. Many physics applications, general purpose diagnostic tools and a physics logger have been developed and tested with a portion of the SNS linac. The overall framework is described, and example applications are shown.

1 INTRODUCTION

The SNS is an accelerator for pulsed, high-intensity neutron production. In addition to EPICS [1] control system, a Java-based software infrastructure called XAL [2] is designed and implemented for general-purpose, high-level accelerator applications. The XAL is a programming framework providing an object-oriented model of an accelerator, interfaces to the SNS control systems for dynamic data and to the SNS global database [3] for static information, interfaces to various external modeling software packages, and a built-in lattice tool (online model) [4] mainly for quick, online calculation. XAL is designed for but not limited to accelerator physics applications. General purpose tools, such as machine protection false analysis, data logging, hardware monitoring services and so on, are also written with XAL. An overview of the XAL structure is described in Section 2. Several applications are shown in Section 3 as examples.

2 XAL STRUCTURE AND DATABASE

The entire software and hardware infrastructure is shown schematically in Fig. 1. A subset of the global database is extracted into an extensible markup language (XML) formatted file. Any XAL application requiring initialization with static data can parse this XML file. The communication between XAL applications and the control systems is through an EPICS Java Channel Access layer embedded in the XAL. A separate XML file called *probe* contains initial beam information for running the online model through a lattice.

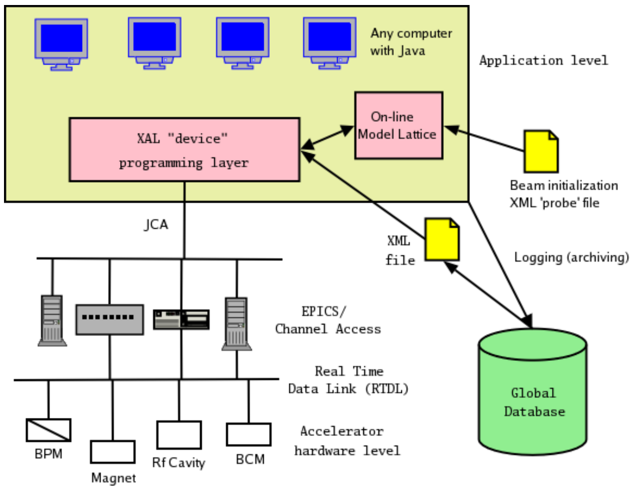


Figure 1: Application software infrastructure

2.1 Accelerator Model

At the heart of XAL is a set of classes describing an accelerator hierarchy as shown in Fig. 2. The accelerator is composed of a set of sequences, which in turn are composed of sets of nodes. The node structure includes classes for common beam-line components such as magnets, RF cavities and diagnostic devices.

Methods are provided throughout the accelerator class structure to easily perform common tasks such as: 1) selecting nodes of a certain type from a sequence, 2) getting or setting magnetic fields in a magnet, or 3) finding the beam position from a diagnostic Beam Position Monitor (BPM). Importantly, the details of the control system connection are hidden from the user. The Accelerator node objects generally correspond to the

* SNS is a collaboration of six US National Laboratories: Argonne National Laboratory (ANL), Brookhaven National Laboratory (BNL), Thomas Jefferson National Accelerator Facility (TJNAF), Los Alamos National Laboratory (LANL), Lawrence Berkeley National Laboratory (LBNL), and Oak Ridge National Laboratory (ORNL). SNS is managed by UT-Battelle, LLC, under contract DE-AC05-00OR22725 for the U.S. Department of Energy.

physical devices actually in the accelerator beam-line, but are not necessarily one-to-one mappings. Other non-beam-line devices, such as magnet power supplies, vacuum gauges, can also be included in a sequence for display or simple calculation purpose.

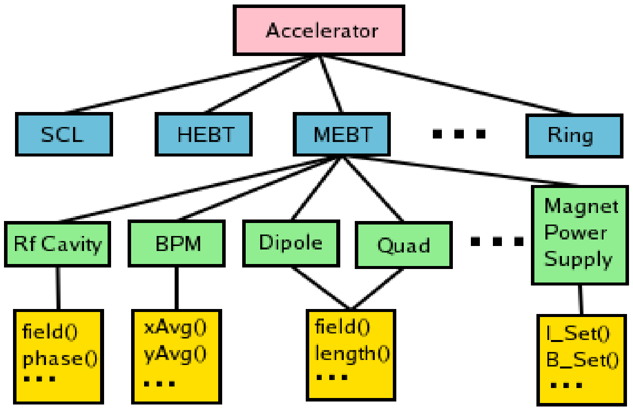


Figure 2: Schematic of the XAL accelerator class hierarchy.

2.2 The Control System Connection

XAL uses EPICS as the underlying control system to communicate with accelerator hardware (see Fig. 1). EPICS communication uses a single "Process Variable" (PV) as the fundamental unit for communication with higher level programs via a protocol called Channel Access. XAL has a Channel class that encapsulates the communication with a PV. As shown in the bottom row of Fig. 2, XAL provides a simple interface to the hardware and hides all the Channel complication from application writers.

2.3 Database connection

XAL provides a hierarchal framework, but the actual objects within this structure need to be initialized. We initialize the accelerator structure using multiple tables in the global SNS relational database [3]. The global database is the central repository containing information about both abstract devices and specific hardware equipment. The database is also configured to store logged physics data taken from the accelerator. For XAL, the abstract device information is of interest. In particular, there are tables defining the sequences within SNS, and devices within each sequence. A beam-line device could be a magnet, diagnostic instrument or RF cavity. A SQL query produces the XAL structure directly.

2.4 Data Correlation

The SNS is a pulsed machine, and will operate at 60 Hz eventually. EPICS will provide a timestamp to each PV from a common timing system. It is important to be able to gather many signals from a common pulse when analyzing beam behavior. An optional data correlation engine in XAL ensures that the event data collected all occurred within a specified time window, usually the beam pulse width.

2.5 Tools

Several general purpose tools are provided that are used in applications and classes. Some of the more extensively used tools are described here.

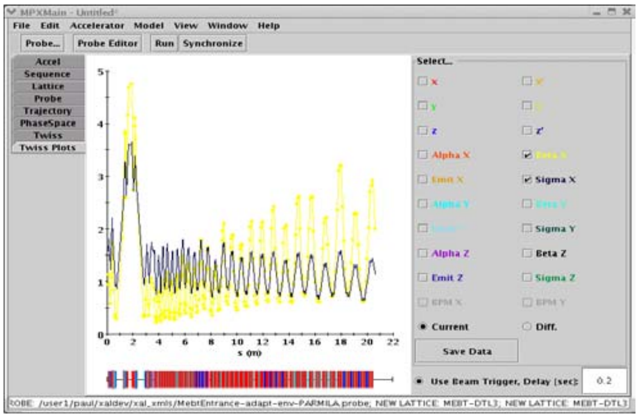
The framework provides a standard user interface design such as common look-and-feel, Java logging and user preference save and restore, and online help in HTML form.

Several mathematics tools, such as optimization, fitting, integration, are included in the XAL. A fast data plotting utility for nearly real-time display is also provided.

In addition to creating pure Java applications to exercise XAL capabilities, scripting interfaces are available. These include Jython and Matlab™ interfaces. In both these cases, no glue code, special wrapper code or build steps are needed; rather the Java classes can be directly imported into the scripting level and used seamlessly along with the scripting language.

2.6 Online model

An important XAL feature is the online accelerator model [4], which allows for on-the-fly calculation of beam parameters, based on machine settings. The online model is loosely coupled with the other part of the XAL. The main components are a lattice (constructed from the XAL accelerator nodes) 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. In the transformation to the lattice view, devices may be split into more than one piece, and drift spaces are added (note that no drift information is stored in the XAL initialization database, only actual device information). Also, a visitor pattern scheme is used to facilitate synchronization of the lattice view parameters, with updates from different sources. It is possible to run the online model outside of XAL, as long as an online model lattice and a probe are supplied. Fig 3 shows an application based on the online model.



ROBE: /user1/paul/xaldev/xal_xmls/MebtEntrance-adapt-env-PARMILA.probe; NEW LATTICE: MEBT-DTL3; NEW LATTICE: MEBT-DTL3

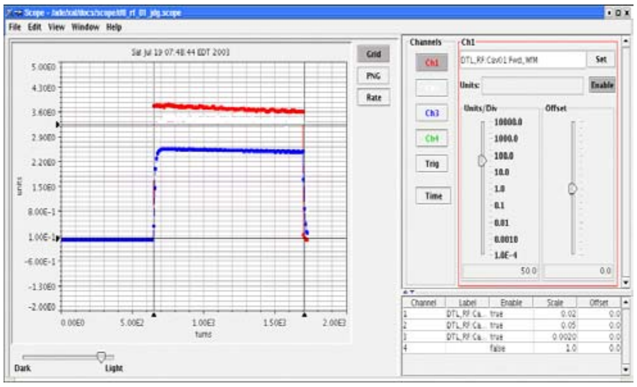
Figure 3: Online model application.

3 APPLICATIONS

XAL tools have been used to create about 20 applications. Most of these applications use Java Swing GUI components. An important feature of the applications is the use of a common framework. This GUI framework provides common look-and-feel for every application. The framework also includes file save/restore, screen capture, error logging, memory usage display, and many other features. Some specific application examples are presented here.

3.1 Scope

There are only minimal provisions for analog display of waveforms in the control room, therefore waveforms are digitized and made available as EPICS PVs. A "digital scope" application is available for viewing the waveforms. An important requirement is to be able to overlay different waveforms from different sources (e.g. RF, BPMs, loss monitors, …). Since each waveform source uses different digitization methods, we require additional information for each waveform PV, describing the time offset of the first element from a common time point, and the time bin size that each array element corresponds to. With this information, waveforms from various sources can be displayed together, with the real time as the x coordinate. The time correlator engine described in Section 2.4 is used to ensure that simultaneously displayed waveforms are from the same machine pulse.

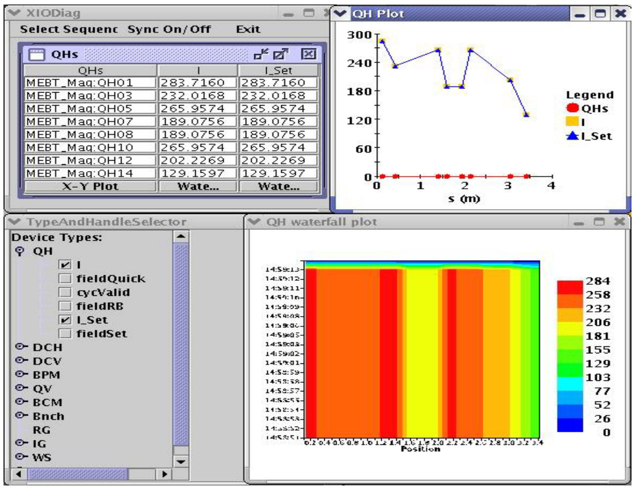


Channel	Label	Enable	Scale	Offset
1	DTL_RF:Ca...	true	0.02	0.0
2	DTL_RF:Ca...	true	0.05	0.0
3	DTL_RF:Ca...	true	0.0020	0.0
4		false	1.0	0.0

Figure 4: Screen snapshot of the Scope application.

3.2 Xio

XIO is a general purpose application to monitor sets of signals. The user first uses the accelerator sequence selection feature (part of the framework) to select the portion of the accelerator to work with. Then device types of interest within the selected accelerator sequence are picked (e.g. BPMs ...). Finally the signal types for the selected device types are picked (e.g. horizontal beam position signal type for the BPMs). A separate table is created for each selected device type within in a tabbed panel and within each table there is a column for each selected signal type. The table cells display the values at a prescribed update rate (typically 1-2 Hz). Options exist to display each column of signals as a live X-Y plot, where the horizontal axis is the distance along the selected accelerator sequence. Also possible is a color "waterfall" display of a signal type's value (rendered via a color mapping) along the beam-line (x axis) and versus time (y axis).



QHs	I	I_Set
MEBT_Mag:QH01	283.7160	283.7160
MEBT_Mag:QH03	232.0168	232.0168
MEBT_Mag:QH05	265.9574	265.9574
MEBT_Mag:QH07	189.0756	189.0756
MEBT_Mag:QH08	189.0756	189.0756
MEBT_Mag:QH10	265.9574	265.9574
MEBT_Mag:QH12	202.2269	202.2269
MEBT_Mag:QH14	129.1597	129.1597
X-Y Plot	Wate...	Wate...

Figure 5: Screen snapshot of the Xio application.

3.3 Virtual Accelerator

To test the framework and applications in the absence of a running accelerator, we rely on an accelerator simulator called the virtual accelerator with the online model as the engine and portable channel access server as EPICS data provider. The advantages of this simulator are model-based simulation, and the same data acquisition interface and the same EPICS process variable (PV) settings as been used on the real machine. A virtual accelerator application using the online model is served as a client of a portable channel access server.

4 CONCLUSION

The basic Java XAL structure is produced, and includes facilities for application programming. A set of initial applications is produced, and used at the initial SNS commissioning. The next development stages include populating the database and developing more applications specific to other beam-lines that will be commissioned in the next several years. Also, in the framework an agent based client/server capability is developed.

5 REFERENCES

[1] http://aps.anl.gov/epics
[2] http://www.sns.gov/APGroup/appProg/xal/xal.htm.
[3] J. D. Purcell, *et al.*, "Initial Experience with SNS Database Applications", ICALEPCS 2003 Proceedings.
[4] C. Allen, et al., "A Novel Online Simulator for High-Level Control Applications Requiring A Model Reference", ICALEPCS 2003 Proceedings.