SNS BEAM COMMISSIONING TOOLS AND EXPERIENCE

A. Shishlo#, J. Galambos on behalf of SNS commissioning team, ORNL, Oak Ridge, TN 37831, U.S.A.

Abstract

The Spallation Neutron Source (SNS) successfully met the primary construction project completion milestones in April 2006. An important ingredient of this successful commissioning was the development and use of software tools. With the increasing digitalization of beam diagnostics and increasing complexity of Integrated Control Systems of large accelerators, the need for high level software tools is critical for smooth commissioning. At SNS a special Java based infrastructure called XAL was prepared for beam commissioning. XAL provides a hierarchal view of the accelerator, is data base configured, and includes a physics model of the beam. This infrastructure and individual applications development along with a historical time line of the SNS commissioning will be discussed.

SNS BEAM COMMISSIONING TIMELINE

Fig 1 shows the commissioning periods for different parts of the SNS accelerator. Commissioning started in the spring of 2002 in LBNL and ended 3.5 years later. SNS beam commissioning occurred in 7 separate stages. Details of the commissioning process are described in [1-3]. The number of commissioning periods (red color on Fig. 1) is more than the number of different types of accelerator lines in SNS, and commissioning of some parts overlapped. Actually, the commissioning could have been done in fewer steps. The reduced number of commissioning periods probably could have saved some time during the transitions from commissioning to installations periods and back. But this type of schedule was chosen intentionally. The approach was iterations of "try-and-learn".

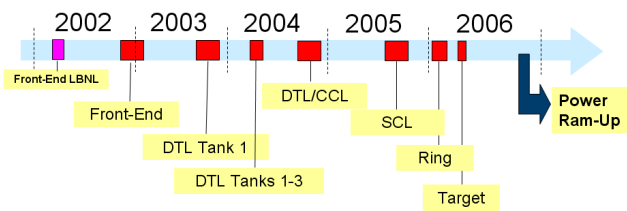

Figure 1: SNS commissioning timeline.

The decision to have beam commissioning activities at an early stage (albeit initially for rather modest lengths of the accelerator) allowed early integrated deployment of major systems such as diagnostics, controls, timing and high level applications. Initially deploying these systems over a rather small piece of accelerator simplified the integration process. Although having many beam commissioning stages complicated planning and installation activities, the benefit was well worth this complication.

The development of commissioning tools started about two years before the first real commissioning period. This period was very important, because decisions made at that time defined the main direction for development of the SNS high level control system.

EARLY DAYS OF COMMISSIONING PLANING

An early decision in the commissioning planning was to develop a high level application programming infrastructure (XAL) [4]. The different potential technologies were reviewed: FORTRAN, MATLAB, EPICS extensions like SDDS (Self Describing Data Sets), Cdev, and Java. Eventually it was decided to proceed with the Java based environment, because it enables integration of a model, GUI, database connectivity, and networking into one package.

The work was divided between the accelerator physics, controls, and diagnostics groups. The control group was responsible for data base development related to XAL. The diagnostics provided the raw signal processing and the necessary physical signals to the XAL applications. To proceed with application and framework development it was decided to create an Application Programming Team inside the accelerator physics group. The team members were responsible for the development a core of XAL infrastructure and the first high level applications that served as examples for others. The team members also served as consultants for fellow accelerator group members. The common practice assumed that one person (not necessarily an application programming team member) is responsible for a particular application beginning with a concept and ending with a central control room version. Having a single person responsible for understanding the underlying physics, writing the application and deploying it with beam reduced debugging turnaround time.

XAL APPLICATION PROGRAMMING INFRASTRUCTURE

All high level application programs that played an important role in the SNS commissioning were implemented within the XAL infrastructure. A key XAL feature is a hierarchal description of the accelerator. The accelerator structure and nodes are configured via a global database. This approach makes creating generic tuning applications that can be applied throughout the accelerator much easier. Since the SNS was commissioned in stages, once a generic application was made to work in an early

#shishlo@ornl.gov

stage, it was available for use in later stages simply by populating the global database configuration for the new stages.

XAL includes a beam transport model [5], which was critical in the commissioning. Use of this model was a key part of the RF cavity tuning, beam steering and focusing, as well as unraveling unanticipated problems. The ability to quickly apply model based analysis and matching was quite useful. The model includes particle and envelope (with space charge) tracking, permiting beam centroid and beam size predictions which were used for comparison with measurements. It also includes longitudinal tracking methods used in the primary linac RF tuning algorithms. All references to the model in this paper refer to this "online model".

Another XAL feature is an application template. This provides a quick-start for GUI application developers, provides a common look and feel for applications, and allows seamless upgrades of all applications. The template made it possible for physicist developers with minimal programming experience to write tuning algorithm applications.

XAL also has a common set of tools that are sharable among applications. These tools include plotting, optimization (used in matching model to beam measurements), database retrieval, signal scanning packages, and correlation (pulse-to-pulse) methods.

In addition to applications, XAL has services. Services are running constantly in the background, and their methods can be called remotely from any XAL application. Since multiple applications may access the same online resources, a dedicated service can serve as an agent for these applications and minimize the system resource requirements. For instance, the PV Logger service can put a "snapshot" of the machine state (from EPICS point of view) into database upon request from any application.

Some of the XAL components that had strongest impact on the SNS commissioning are discussed below.

Accelerator Model

At the heart of XAL is a set of classes describing an accelerator hierarchy as shown in Fig. 2. This hierarchy is a mirror image of the real accelerator logical and physical structure. The accelerator model 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. In the present state XAL components communication with the real machine is based on the EPICS network data exchange protocol. To avoid a direct dependency on future EPICS modifications XAL has a layer of abstract classes providing an access to the EPICS channel access library.

The Accelerator node objects generally correspond to the physical devices actually in the accelerator beam-line, but are not necessarily one-to-one mappings. For instance, several magnets can have one power supply. 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.

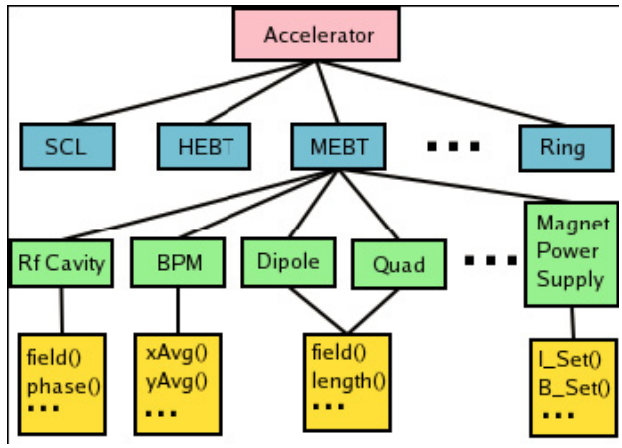

Figure 2: XAL accelerator model class hierarchy.

Online Model

An important part of XAL is the online accelerator model, which allows for on-the-fly calculation of beam parameters, based on live machine or user defined 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 model nodes) and a probe (describing the beam, and how it is to be treated in nodes). 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. There are several possible sources for synchronization of an instance of the online model. It could be the design lattice; live machine data; PV logger machine snapshot, user defined 'what-if' magnet and RF settings; or a combination of sources.

The XAL online model uses transport matrices to track the probe through the accelerator lattice. When the probe represents an envelope of the bunch in 6D space, an algorithm taking account linear space charge forces can be added to the tracking. The relative simplicity of the XAL model makes it very fast and allows its usage inside interactive optimization routines for tasks like orbit correction, matching etc.

Database

The SNS global database has a schema design suitable for all the static information needed by various XAL applications. Although the global database is not part of the XAL framework, it is configured based on the XAL requirements. A database query application can generate an accelerator file whenever there is some beam-line device change in the database. This XAL accelerator file is the sole source for the accelerator model instantiation, and it has an XML structure similar to one shown in Fig. 2. The existence of an intermediate initialization file between the model and the database is very convenient when you do not have a permanent open access to a database.

Customized online data including physics related machine snapshots are also stored in the database via the PV logger. Many applications can also take PV logger data for offline analysis.

Tools

Many general purpose tools have been written, which are shared between applications. A small portion of all the available tools are described here. The SNS is a 60 Hz pulsed device, and each signal has a timestamp attached, corresponding to the start of the pulse. A correlation tool allows collection of groups of signals with comparable timestamps. It has many features including triggered acquisition to allow sets of data to be grabbed only when some specified external criteria is met.

A data table structure provides simple database like functionality. This structure provides more capability than the built in Java collections (e.g. database like querying), yet is much simpler to implement than construction of a real set of database tables.

A communication layer is available for client-service needs. It uses XML-RPC for inter-process data exchange and uses Rendezvous for discovering subscribers and publishers. Knowledge of the details of Rendezvous and XML-RPC are hidden from the user.

The XAL optimization package is a product of a long evolution of such packages during the XAL development. Currently it is the third generation of optimization packages in XAL. It is based on such advanced conceptions as a pool of algorithms, a dynamical competition of algorithms, multiple objectives of the optimization, collection of optimization process stoppers etc. The XAL optimization package has no upward connection to other XAL classes, and it can be freely used outside of XAL.

Additional tools include plotting, database connection, XML data parsing, and external modeling tool support.

XAL Application Framework

In the early days of XAL development, each application developer created his own JFrame with menus, toolbars, and all standard functionalities like open, save etc. This approach meant duplicated efforts, a different look and feel for each application, and maintenance difficulties. To avoid these drawbacks, the XAL Application Framework was created. It provides the common look and feel, basic functionalities, and some XAL related services freely available to all application developers.

CONCLUSIONS

To sum up our experience from the SNS commissioning we want emphasize several things that were successful: an early and multiply-staged commissioning approach; an iterative approach for commissioning tools development; using physicists (i.e. commissioners) to write applications (need a core group of “mentor” programmers). Regarding the XAL development: choosing Java as a development platform; initialization files created from a database; the XAL online model; the XAL application framework; scripting (Jython/JRuby).

Of course, there were also a number of failures and bad choices made during the development: too much SNS specifics in XAL; lack of documentation; usage of a commercial plotting package in free source software (XAL).

ACKNOWLEDGMENT

Research on the Spallation Neutron Source is sponsored by the Division of Materials Science, U.S. Department of Energy, under contract number DE-AC05-96OR22464 with UT-Battelle Corporation for Oak Ridge National Laboratory.

REFERENCES

[1] N. Holtkamp, “Commissioning Highlights of the Spallation Neutron Source”, Proceedings of the 2006 EPAC, Edinburgh Scotland, http://accelconf.web.cern.ch/AccelConf/e06/PAPERS/MOZAPA02.PDF.

[2] A. Aleksandrov, et.al., “SNS Warm Linac Commissioning Results”, Proceedings of EPAC 2006, Edinburgh, Scotland, http://accelconf.web.cern.ch/AccelConf/e06/PAPERS/MOPCH127.PDF.

[3] M. Plum, et.al., “SNS Ring Commissioning Results”, Proceedings of EPAC 2006, Edinburgh, Scotland, http://accelconf.web.cern.ch/AccelConf/e06/PAPERS/MOPCH131.PDF.

[4] J. Galambos, C.K. Allen , C. Chu, S. Cousineau, V. Danilov, J. Patton, T. Pelaia, A. Shishlo, “XAL Application Programming Structure”, Proc. of the Particle Accelerator Conf., Knoxville TN, 2005, http://accelconf.web.cern.ch/AccelConf/p05/PAPERS/ROPA001.PD

[5] C. K. Allen, C. A. McChesney, C. M. Chu, J. Galambos, T. Pelaia, “A Novel Online Simulator for Applications Requiring a Model Reference”, http://accelconf.web.cern.ch/AccelConf/ica03/PAPERS/WE116.PDF.