EPICS and its Role in Data Acquisition and Beamline Control

T. M. Mooney, N. D. Arnold, E. Boucher, B. K. Cha, K. A. Goetze, M. R. Kraimer, M. L. Rivers, R. L. Sluiter, J. P. Sullivan, D. B. Wallis

Advanced Photon Source, Argonne National Laboratory, Argonne IL, 60439

Abstract. Beamline-control and data-acquisition software based on EPICS (a tool kit for building distributed control systems) has been running on many Advanced Photon Source beamlines for several years. EPICS itself, the collaborative software-development effort surrounding it, and EPICS-based beamline software have been described previously in general terms. This talk will review and update that material, focusing on the role EPICS core software plays in beamline applications and on the effects of a few defining characteristics of EPICS on the beamline software we have developed with it.

BACKGROUND AND EPICS OVERVIEW

EPICS (Experimental Physics and Industrial Control System) (1) is a software tool kit for building distributed, client-server control systems, and the focus of a large, international collaboration of software developers. Collaborators include developers of control systems for particle accelerators, telescopes, synchrotron-radiation (SR) beamlines, and large experimental physics detectors, among others. SR facilities at which EPICS-based beamline software is under active development include the Advanced Photon Source (APS), Berliner Elektronenspeicherring-Gesellschaft für Synchrotronstrahlung m.b.H. (BESSY II), and the Swiss Light Source.

At the APS, EPICS was chosen as the standard base for beamline-control and data-acquisition software for many reasons, but the most potent were the desire to spread the costs of development and debugging as widely as practical, and the hope that software standardization would allow users, hardware, and techniques to move easily from one beamline to another. EPICS is a very effective vehicle for collaborative beamline-software development, and it provides excellent support for moving hardware and techniques from one beamline to another.

The central organizing idea in EPICS is the *record*, a collection of named values (*fields*) qualified by attributes, and a set of functions that operate on those fields. An example is the *Analog Output* record, whose fields include an analog value; the units in which the value is expressed; limits on the value; fields describing how the value should be displayed to the user; fields and attributes describing when the record functions should be executed; fields for the private use of the record functions; fields used by EPICS core software to manage the execution of those functions; a field naming the device-support module that is to convey the analog value to hardware or to another record; and fields for the private use of that device-support module.

Records can stand alone or be linked into collections, called *databases*, and the links can convey values, cause execution, or do both. A database is effectively a high-level program and, since records in one database can link to records in another database, a collection of databases can be viewed as a still higher-level program. The beamline user can operate in this software layer because EPICS links are modifiable at run time, and because links are specified by strings that a client can read and write.

At the highest levels are client programs, most of which run on workstations and communicate with server software via the network, using an EPICS-supplied mechanism called "channel access". Software running in the VME crate can also communicate with other crate-resident software via channel access. The scan software described below is an example of a module that runs in the crate and behaves like a client toward lower-level software, such as an optical table (which in turn is a client of the motor records it controls), and like a server toward higher-level software, such as the user interface.

These four software layers—device-support, record-support, database, and client—provide good opportunities for collaborative development: EPICS allows modules in all of these layers to be added and removed from a system with minimal effects on unrelated software, the layers are cleanly defined, and they require different skills and knowledge. To program effectively in the device-support layer, for example, a developer must understand hardware interfacing, interrupts, etc., but need not know anything about the end use to which the device-support will be put. Also, this developer need not know about networking, user-interfaces, graphics, etc. To write an application-specific record type, a developer typically must have a thorough understanding of EPICS and of the application, and must be a competent programmer, but need not understand the VME backplane, or be an expert in TCP/IP, user interfaces, etc. To program in the database layer, a developer must understand records, EPICS links and execution mechanisms, and the application, but does not need an extensive programming background. Client developers must understand channel access, and event-driven programming, but require no knowledge of hardware interfacing.

Much of the collaboration involved in producing beamline software has been unconscious or at least undirected. For example, an accelerator-controls developer at the

Continuous Electron Beam Accelerator Facility (CEBAF) wrote support for a waveform generator, an accelerator-controls developer at APS wrote support for a digital I/O board, one of us wrote a database using this digital I/O module to control a piezo actuator, another wrote support for a multichannel scaler board, and another modified the waveform-generator software and collected other modules into a system that can acquire tomography data at up to 10^4 pixels/second. The parameters that control the tomography software (and other beamline software) are managed by a save/restore mechanism developed by an EPICS core developer at the Los Alamos National Laboratory (LANL), and modified by us and by a developer at the Stanford Linear Accelerator.

Though undirected, this kind of collaboration is not unintentional. Most EPICS developers expect their code to be re-used by others, and this expectation colors their approach to problem solving. It encourages generic solutions to classes of problems instead of specific solutions for narrow requirements. It also provides an incentive to produce robust code. In effect, collaboration of this kind is a form of peer review.

RECENT DEVELOPMENTS IN BEAMLINE SOFTWARE

Most of the main features of current EPICS-based beamline software existed, at least in prototypical form, when the software was last described in this forum (2). Where practical, improvements were made in ways that required minimal changes in the user interface, and in existing client software.

Scan support: The scan software uses a new mechanism for determining when operations initiated by a scan (e.g., motor motion and scaler counting) have completed. Previously, all scannable positioners and detectors maintained flags with which they signalled completion, and the completion detector was simply a logical OR of all the flags. This mechanism was slow and easily fooled by network latency, and in some cases it required user involvement. EPICS has the ability to trace execution in a complex, distributed database, and we discovered how to use this mechanism in the scan software. The new completion detector is much more robust, roughly eight times faster, and does not require (or allow) any user involvement. However, it requires some care from the developer, because not all of the possible execution paths in EPICS are traced. Most of the records and databases in our software required some modification to conform to execution-trace requirements.

The scan software now writes data directly to the file server, instead of relying on a separate client running on the workstation. This simplifies the handshaking required between acquisition and storage by removing network latency, and it separates responsibilities for display and storage, improving performance for both. We are currently developing software to translate scan-data files into the NeXus (3) format.

Motor support: Device-support software for motor control was rewritten to minimize the development required to support new controllers. Software common to all controllers was separated out and the interface to controller-specific modules is being standardized and documented. One new device-support module that is still being refined allows the motor record to act as a front end for a virtual motor, such as the insertion-device energy or the tilt of an optical table. This software will allow client applications that already know how to use the motor record to control nearly anything else on a beamline by treating it as a motor. Thus far, we have applied this technique only to slit-control software.

Scaler support: The method previously used to handle interrupts from scaler hardware interfered with EPICS' execution-trace mechanism. The method used to fix this problem also was used to replace the old "autocount" feature (counting for display only, when the scaler would otherwise be idle) with a more robust version that does not interfere with or delay commands from clients, such as the scan software.

Multichannel analyzer/scaler support: The multichannel-analyzer (MCA) record was originally developed as a front end for device-support software contributed by Canberra and modified to run under the VxWorks operating system. This software allowed a single spectrum to be acquired from the Canberra Acquisition Interface Module (AIM). Recently, the record was modified extensively to support new techniques and new hardware, and new software was written to support Struck multichannel scalers. The new software supports multiplexed ADC inputs, dedicating an MCA record to each signal. This technique was carried over into device support for multichannel scalers. Support for time-resolved spectroscopy was also added to the MCA record. Currently supported MCA hardware can acquire spectra at around 20 Hz.

Support for Canberra Instrument Control Bus (ICB) modules was also developed. Currently, a spectroscopy amplifier, a spectroscopy ADC, a high-voltage power supply, and a triple-channel analyzer can be controlled with EPICS using an AIM module as the ICB master.

An MCA/MCS display and control program was written in IDL. The program provides for spectrum calibration from a database of x-ray energies, manages regions of interest, and can store MCA data automatically when acquisition completes.

Optical table support: Optical-table software rotates the table about a user-specified point and/or translates that point, and can recover table angles and translations ("virtual motors") from real motor positions. (This capability is required because we allow clients other than the table record to control table motors, and because we must calculate virtual-motor limits from real-motor limits.) Previously, all calculations were done to first order in small quantities because this simplified the real-to-virtual equations so that we could solve them analytically. In the new software, real-to-virtual and virtual-to-real equations are exact, and several new table geometries are supported.

Parameter save/restore: Previously we had no way to automatically preserve user-modifiable parameters, such as motor speeds, through a crate reboot. An EPICS core developer at LANL solved most of the problem, and the most recent autosave/restore software is robust enough to survive a crate reboot that occurs while the save set is being written.

Monochromator support: Support for dispersive double-crystal and spherical-grating monochromators has been added. The dispersive double-crystal monochromator software allows for crystals of different species and orientation, and has several modes of operation: 1) both crystals at the Bragg angle for a specified energy; 2) first crystal fixed while the second crystal rotates; and 3) both crystals rotate to compensate for a change in the incident-beam angle. The spherical-grating monochromator software supports multiple gratings with different line densities and radii of curvature.

Multistep measurement support: Previously we provided no convenient way for the user to program multiple-step measurements, such as those required in dichroism experiments. Recently developed software allows up to four measurement steps each involving a user-programmed sequence of operations followed by the triggering of detectors and, on detector completion, acquisition of four signals followed by user-programmed calculations. The software can be used as a detector in a scan.

Links: Previously the only EPICS links that could be modified at run time were the "dynamic links" implemented as part of the scan software. Now all EPICS links can be modified at run time, and the new implementation does not funnel all retargetable link operations through a single task, as the previous implementation did. We modified the old dynamic-link software so that it can invoke EPICS' execution-tracing mechanism, and issue a callback when execution completes.

Run-time expression evaluation: EPICS includes software to evaluate mathematical expressions entered by the user at run time. We have added new operators to this software, extended it to handle string expressions, and extended the EPICS sequence and calculation records to handle strings as well as numbers. This allows users to program simple serial and GPIB devices at run time, and to integrate those devices into the rest of their software so they can be scanned or involved in feedback loops. It also allows an EPICS database to retarget its own links in response to a user command.

Support for IndustryPack Hardware: IndustryPack (IP) is a mezzanine bus with specifications reasonably well suited to an SR beamline's need for miscellaneous I/O (e.g., DAC's, ADC's, serial communication). Previously, this hardware was supported by Hideos, but that system provided little diagnostic support, and was inconvenient to build and maintain. Hideos became a prototype for the EPICS Message Passing Facility (MPF), which has recently matured into a deployable solution for running beamlines. MPF runs under VxWorks, both in the main EPICS processor and in the satellite IP carrier board. Legacy software that used Hideos is easily ported to MPF.

Effect of EPICS on Beamline Software

The principal effect of EPICS on beamline software developed with it has been the preference for general-purpose solutions mentioned earlier. This approach supports more applications than would a narrow-focus approach, but the support for any one application can be more complicated to develop and use than special-purpose software would have been.

Until recently, only software running in the VME crate (server-side software) could publish names for use by other EPICS software. This channelled most APS-supported effort into server-side development, because this software can be used on most APS beamlines, and it can be driven by other server-side software as well as by clients.

Another preference for server-side development is an effect of EPICS' support for shared local/remote beamline operation. It's not yet clear whether users generally will take advantage of this capability, but we didn't want to make the choice for them by throwing the capability away. To preserve it, we've kept as much state information as practical on the server side, where it's available to both local and remote users.

ACKNOWLEDGEMENTS

Many software developers have contributed directly or indirectly to EPICS-based beamline software, and hundreds of APS users have helped to debug it.

Work supported by the U. S. Department of Energy, Basic Energy Sciences, Office of Science, under contract no. W-31-109-Eng-38.

REFERENCES

1. L. R. Dalesio, M. R. Kraimer, A. J. Kozubal, "EPICS Architecture," in Proceedings of the International Conference on Accelerators and Large Experimental Physics Control Systems, Tsukuba, Japan (1991).

2. Mooney, T. M., et al., "Beamline Control and Data Acquisition Software," Rev. Sci. Instrum. 67 (9) CDROM 1996.

3. http://www.neutron.anl.gov/NeXus/