ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-THPG21

ACCELERATOR CONTROL SYSTEM SOFTWARE AT LANSCE: VISION AND STRATEGY FOR IMPROVEMENT AND MODERNIZATION*

E. Westbrook†, S. Baily, S. Elliser, D. Fratantonio, M. O'Connell,
M. Pieck, H. Watkins, D. Zimmermann
Los Alamos National Laboratory, Los Alamos, NM 87545, USA

Abstract

The LANSCE accelerator is an 800 MeV linear accelerator delivering beams for more than fifty years. As it has aged, maintenance and upgrades to its control system software components have become challenging and often deferred due to operational and schedule constraints. As a result, we have a wide variety of new and old software, difficult to re-use, with a large staff burden. Data is stored in redundant sources, inconsistent formats, and outdated technology. Multiple tools exist for the same tasks. Some production software is updated without proper processes. We describe our approach to modernizing LANSCE control system software with proper development processes. We consider reduction of diversity, redundancies, data sources. Migration to modern technologies is also discussed. We explore the possibility of language standardization and describe our database implementation and other future plans. Lifecycle management is also considered. This years-long effort will utilize a risk-based strategy to address the most urgent issues while also ensuring steady progress, ultimately resulting in a coherent and maintainable suite of control system software.

BACKGROUND: TECHNICAL DEBT

Technical debt: The implied costs of future rework resulting from cumulative expedient but limited choices.

RISK AND PRIORITY ASSESSMENT

Table 1 lists the areas to be discussed herein along with general estimations of risk and assignment of priority.

Table 1: Risk and Priority Assessments

Area	Risk	Timeline
Engineering Process	High	<3yrs
EPICS	High	<3yrs
Control System Network	High	<3yrs
Data Management	Med	<6yrs
Language Diversity	Med	<10yrs
Lifecycle Management	Med	<5yrs
Server Architecture	Med	<2yrs
Linux OS Diversity	Med	<2yrs

* This work was supported by the U.S. Department of Energy through the Los Alamos National Laboratory. Los Alamos National Laboratory is operated by Triad National Security, LLC, for the National Nuclear Security Administration of U.S. Department of Energy (Contract No. 89233218CNA000001), LA-UR-24-24831.
† westbrook@lanl.gov

SOFTWARE ENGINEERING PROCESS

In recent years, LANSCE's [1-5] control system software has undergone dramatic improvements and modernizations, with more planned for upcoming years.

Technical Debt and Risks

- Previously, code was developed *in-situ* within active production release directories, posing risk to operations, collaboration, and revision management.
- Few controls were in place to ensure that development processes were repeatable, effective, or even followed.
- Software was developed using SVN rather than git, posing technical as well as staffing risks.

Vision and Strategy

Many improvements have been made in recent years, including migration to git, introduction of CI/CD, and process formalization. Future efforts are intended to include:

- CI/CI pipelines for build/test/deploy of compound production software which is not yet covered;
- Comprehensive peer code review process;
- Expedited exception process for on-call emergencies;
- Automated enforcement of process requirements;
- Transfer of software currently managed by other groups to the control system software team and its processes.

EPICS

Technical Debt and Risks

LANSCE relies on the EPICS control system framework. In recent years, we have enhanced EPICS with tagging and filtering functionality unique to LANSCE, in enhancements we call "DAR." Unfortunately, the combination ("DAR EPICS") was developed without preserving distinct separation of the two. We now risk being locked out of bugfixes and enhancements in new upstream EPICS versions. We also risk the inability to contribute enhancements to the community.

Vision and Strategy

- At a minimum, and before anything else, the DAR enhancements must be separated into (at least) one clean and distinct DAR "patch set". This provides the critical ability to build an identical DAR EPICS code product by applying this DAR patch set back on to the original (or any other) EPICS version.
- For goodwill and maintainability, it would be good to further refactor out individual DAR enhancements and

ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-THPG21

contribute some or all of them back to the EPICS community. DAR would then be reduced by these accepted components, since they would then be in EPICS. Ideally, this could reduce DAR to zero in this way, but that is unlikely: The upstream community is not expected to accept all such contributions, even if some.
- Therefore, we must prepare to maintain DAR indefinitely, by rigorously curating and maintaining the patch set independently. With each upstream EPICS upgrade cycle, we then update and reapply.

CONTROL SYSTEM NETWORK

Technical Debt and Risks

The controls software team at LANSCE also operates the control system network. The debt at issue includes:
- Insufficient monitoring and capacity analysis;
- Past-end-of-life network switching equipment;
- Past-end-of-life firewall equipment;
- Excessive single-points-of-failure;
- Lack of redundant network paths;
- Insufficient use of routing in lieu of flat numbering;
- Improperly numbered network blocks;
- Significant detritus in network configurations.

Vision and Strategy

The following improvements are planned and/or currently in progress:
- New distribution switches;
- New and redundant core switches;
- A multi-year rotating switch replacement schedule;
- Monitoring with vendor and in-house tools;
- Routed traffic architecture;
- Leveraging of cable plant using VLANs;
- Configuration clean-ups and curation.

DATA ACCESS AND MANGEMENT

Technical Debt and Risks

No Single Source of Truth Different solutions to different problems have led to the creation of different and sometimes contradictory data authority regimes. Multiple processes work with the same data but store it differently in different places. Without a "single source of truth," ambiguous dependencies by other programs results in risks of complication and fragility.

Inconsistent Access Functions Functions to access data are also numerous, redundant, scattered, and inconsistent. In earlier days, language diversity could excuse some of this, but even within the same language we find multiple implementations which have not been reconciled nor shared. This results in risks of inconsistent and even dangerous software behavior.

Vision and Strategy

Three-Tier Architecture Providing the data and logic tiers from the classic three-tier software architecture model will maximize code re-use and simplify software applications.

Data Tier: Central Relational Database Service To improve data consistency and availability, work on a centralized relational database service is underway. Within this database service, common schemas are being created. Processes that currently create data in scattered locations will update the central database at the same time. As the database matures, remaining programs can cut over and use it, one by one, until the old and scattered data stores become unnecessary.

Logic Tier: Centralized API Service In front of the new centralized database service, a language-agnostic API service is being created. This API service can expose common library functions via multiple RPC access methods such as REST, using common data formats such as JSON. By exposing functions through protocols and standards instead of language-specific libraries, software components and programs of all languages can invoke API functions to access data infrastructure in their own natural ways. Thus, we dramatically reduce duplication of function and effort regardless of programming language.

LANGUAGE DIVERSITY

Technical Debt and Risks

Over the years, the list of software programming languages in use at LANSCE has only grown. Supporting too many different programming languages brings risks in hiring and training new engineers, and risks lost time with even seasoned engineers due to context switching.

Vision and Strategy

Python Python's future at the heart of LANSCE is bright. Its language features and performance are versatile and robust, its body of available libraries is vast, it is well suited for scripts and large programs alike, and its popularity bodes very well for recruiting talent.

For these reasons, we have already formed a Python collaboration and learning task force, and we have launched a multi-year project tasked with identifying and converting legacy programs in other languages to Python.

C, C++, Java Because LANSCE relies heavily on the EPICS control system framework, and because of their value in Linux systems generally, C and C++ will probably be with us indefinitely.

LANSCE also has a significant Java codebase and will therefore also be with us for quite a while. We intend to opportunistically migrate Java code to Python whenever presented with significant rework or end of life situations.

FORTRAN, Tcl/Tk While the bulk of our FORTRAN went away when VAX/VMS did, the rest will be eliminated at the completion of a separate smaller project currently underway.

Tcl/Tk is a difficult language for the unfamiliar. Recruiting talent for Tcl/Tk is also very difficult. It also has known dependency issues on newer Linux distributions.

ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-THPG21

Therefore, we are well underway in converting Tcl/Tk code to Python on a triaged basis.

LabVIEW LabVIEW is a vendor requirement on many of our hardware devices. However, we generally limit use of in-house LabVIEW software to intermediary functions between EPICS and the hardware devices in question.

SERVER ARCHITECTURE

Technical Debt and Risks

As capability needs increased, so did the number of standalone computing servers, but so also did the complexity of service components and infrastructure within and among those servers. Locations, dependencies, interconnections, and lifecycle status of services became difficult to manage and track, resulting in the risk of ever-increasing infrastructure detritus.

Vision and Strategy

We have deployed a redundant pair of hypervisor servers, and created a structure for small virtual machines each representing a single service (or a small number of closely-related services). A multi-year project is well underway to migrate services to purpose-specific virtual machines as determined by urgency and opportunity. As services migrate to this virtualization environment, the older metal servers can be retired and salvaged.

LINUX OS DIVERSITY

Technical Debt and Risks

While a Linux-proficient engineer rarely has difficulty adjusting to different packaged Linux distributions (i.e. "distros"), there are disadvantages to excessive variety. Currently at LANSCE we have deployments of RHEL5, RHEL6, RHEL7, RHEL8, Gentoo, and Rocky 8 on our servers and workstations. As a result, software and scripts risk duplicated and "fine-tuning" for each distro in order to work the same way everywhere.

We also have embedded implementations using VxWorks and RTEMS.

Vision and Strategy

An effort to transition all servers and workstations exclusively to Rocky 8 has been underway for over a year now and is expected to be 100% complete during the upcoming fiscal year. We have also developed an extensive Ansible library that allows us to update, spin up, or restore any target system from disaster to full operation very rapidly. Furthermore, the Ansible configurations are all revision controlled in git for comprehensive configuration management and auditability.

Due to vendor lock-in and hardware dependencies, VxWorks and RTEMS will probably continue to be with us for some time. We are exploring paths toward eliminating VxWorks and upgrading RTEMS, and ideally, we would like to consolidate our embedded solutions on a single Linux variant. We currently favor RT Linux for this purpose with RTEMS as a fallback.

LIFECYCLE MANAGEMENT

Technical Debt and Risks

Software lifecycles at LANSCE have typically meant upgrade projects when a specific need is imminent.

- Linux OS upgrades: Each major Linux release upgrade brings changes in library components, configuration, processes, and more. These changes risk breakage of in-house software that depends on certain package or library versions.
- LabVIEW Updates: New LabVIEW releases often do not come with hardware support we need for existing equipment but do come with other features we would like. We risk maintaining multiple versions simultaneously (and must do so). Some LabVEW versions that we must maintain also require supporting software i.e. Silverlight, which is beyond end of support since 2019.

Vision and Strategy

- Linux: Thanks to other efforts including Ansible deployment and reduction of distro variety, it is now possible to plan for Linux OS upgrades more regularly with more predictable schedules than in the past. A reasonable goal seems to perform a major upgrade every two years.
- LabVIEW: There's probably not much we can do about LabVIEW, for the reasons mentioned above.

REFERENCES

[1] Los Alamos National Laboratory, LANSCE: Unique capabilities for science and national security, https://discover.lanl.gov/news/0610-lansce

[2] Los Alamos Neutron Science Center, https://lansce.lanl.gov/about/history.php

[3] M. Pieck, C. D. Hatch, J. O. Hill, H. A. Watkins, and E. E. Westbrook, "LANSCE Control System's 50th Anniversary", in *Proc. NAPAC'22*, Albuquerque, NM, USA, Aug. 2022, pp. 482-485. doi:10.18429/JACoW-NAPAC2022-TUPA62

[4] M. Pieck, C.D. Hatch, H.A. Watkins, E.E. Westbrook, "Ongoing Improvements to the Instrumentation and Control System at LANSCE", in *Proc. ICALEPCS'23*, Cape Town, South Africa, Oct. 2023, pp. 979-985. doi: 10.18429/JACoW-ICALEPCS2023-WE2BCO03

[5] M. Pieck, C. Hatch, E. Westbrook, and H. Watkins, "LANSCE's instrumentation and controls system modernization", in *Proc. IPAC'23*, Venice, Italy, May 2023, pp. 4783-4786. doi:10.18429/JACoW-IPAC2023-THPL133