STRUCTURING AN EPICS SYSTEM TO OPTIMIZE RELIABILITY, PERFORMANCE AND COST*

M. Bickley, K. S. White
Jefferson Laboratory, Newport News, Virginia

ABSTRACT

Thomas Jefferson National Accelerator Facility (Jefferson Lab) uses EPICS as the basis for its control system, effectively operating a number of plants at the laboratory, including the Continuous Electron Beam Accelerator (CEBA), a Free Electron Laser (FEL), Central Helium Liquefier, and several ancillary systems. We now use over 200 distributed computers running over a complex segmented network with 350,000 EPICS records and 50,000 control points to support operation of two machines for three experimental halls, along with the supporting infrastructure. During the 10 years that EPICS has been in use we have made a number of design and implementation choices in the interest of optimizing control system reliability, performance, security and cost. At the highest level, the control system is divided into a number of distinct segments, each controlling a separate operational plant. This supports operational independence, and therefore reliability, and provides a more flexible environment for maintenance and support. The control system is relatively open, allowing any of the 300 account holders to look at data from any segment. However security and operational needs mandate restricted write access to the various control points based on each user's job function and the operational mode of the facility. Additionally, the large number of simultaneous users, coupled with the amount of available data, necessitates the use of throttling mechanisms such as a Nameserver, which effectively reduces broadcast traffic and improves application initialization performance. Our segmented approach provides natural boundaries for managing data flow and restricting access, using tools such as the EPICS Gateway and Channel Access Security. This architecture enables cost optimizations by allowing expensive resources, such as Oracle, to be effectively shared throughout the control system, while minimizing the impact of a failure in any single area. This paper discusses the various design decisions, and how Jefferson Lab has chosen to balance the often competing interests associated with operating and supporting the control system for a large machine.

A BRIEF HISTORY OF EPICS AT JEFFERSON LABORATORY

In 1995, Jefferson Lab started using EPICS to control CEBA. This choice was made in the midst of accelerator commissioning a difficult time to make such a transition. The control system then in use, named TACL, had a demonstrated lack of scalability. It was clear to the Controls Software Group that it would not be possible to effectively control the completed accelerator using TACL. For that reason, despite the risks associated with such a profound change at such a critical time, we made the decision to use EPICS. The timing of that decision resulted in implementation choices with some long-term impacts that made the lab's technical staff particularly sensitive to the tradeoffs associated with control system optimizations.

PERFORMANCE

The most straightforward of the control system design choices are related to performance. The typical solution to a performance limitation involves upgrading the resource that is a limiter. This can be upgrading an overloaded multi-drop network to a switched network fabric. The solution may involve adding memory to processors that have high levels of RAM utilization. For control systems that have oversubscribed control CPUs, it can mean installing additional processors to spread the

*NOTICE: This work was supported by DOE contract DE-AC05-84ER40150 Modification No. M175, under which the Southeastern Universities Research Association (SURA) operates the Thomas Jefferson National Accelerator Facility.

workload. These obvious solutions depend on the availability of financial resources for the necessary procurements.

Performance bottlenecks are most common in situations where channel densities are high. When Jefferson Lab first started using EPICS to control its accelerator, during machine commissioning, we had a ratio of EPICS records to front-end computers that was higher than had ever been attempted. The accelerator was controlled with higher CPU loads and greater memory utilization than had been attempted at other EPICS sites, testing the scalability of the control system in new ways. Those challenges presented new opportunities for the controls group. Because there was no possibility of immediate relief in the form of additional funds for hardware procurements, alternative solutions had to be found.

The principal problem we faced, which had an obvious impact on users of the control system, was excessive CPU utilization. The communication tasks on the front-end processors, being lower priority than the control tasks, were sometimes starved for time. The result was inconsistent network communication and slow connection time. Our environment was very susceptible to the kind of CPU loads that the EPICS network protocol, channel access (CA), imposes for initiating channel connections. Recognizing this, we developed an alternative mechanism, a nameserver, which bypasses the standard connection initiation and eliminated this distributed load from the lab's front-end computers [1][2]. Figure 1 shows the CPU idle time on a front-end computer, illustrating the performance improvement achieved using the nameserver. On a lightly loaded system, heavy connection loads without the nameserver consume 5-15% of the total CPU time, over a time span of 10 to 15 seconds. Because some of the lab's processors were running at a CPU load of 80%-90% this relatively modest enhancement paid off with significantly better performance of other data clients, making their behavior more consistent and reliable.

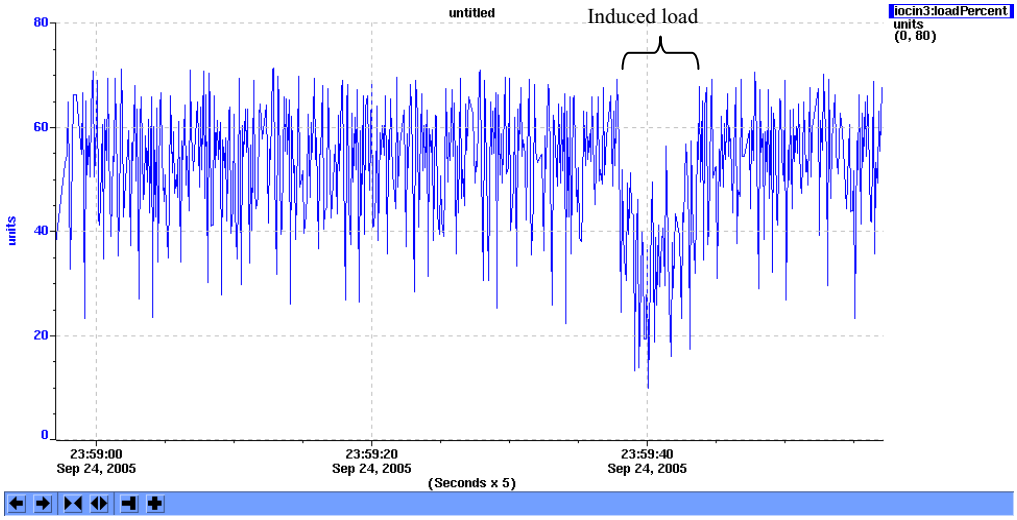

Figure 1: A plot of CPU idle time, showing the load induced on a front-end computer by the standard EPICS connection protocol

In the time since the nameserver was required for successful operation of the accelerator, we have installed additional IOCs, and we have upgraded existing processors from MC68040-based to PPC705-based single-board computers. While the nameserver is no longer required for effective operation of the control system we continue to use it. It has proven to be extremely reliable, and it continues to serve as a load-balancer. It eliminates spikes in both network and CPU load during some connection-intensive operations.

The channel access gateway, a tool developed at BESSY, provides another method to ensure stable performance of the control system. Jefferson Lab has installed four gateways to help isolate control functions from the impact of other operational plants. Just as the nameserver does, gateways eliminate load on front-end systems. In the case of gateways this is because each gateway serves as a connection filter and data concentrator. Taken together, the gateways and nameserver enhance our ability to

tightly manage the impact of data clients on the front-end systems, and therefore ensure consistent and reliable performance from these systems.

COST

The Controls Software Group at Jefferson Lab has made a number of design decisions in an effort to reduce the cost of the control system while continuing to focus on maximizing control system availability. Most of the important cost-related decisions concern the structure of the control system. One set of choices relates to the organization of functionality of the low-level controls on front-end computers. The second category of choices is associated with the structure of the network and back-end computing resources available to control system users.

Low-level Controls

When organizing the low-level controls portion of the system there is a fundamental tension between two poles of organization: consolidation and isolation. By consolidating the controls of multiple disparate devices on one front-end computer, those controls can share the relatively expensive front-end hardware. This can enable savings in cabling costs, because the software for a device can be located in a way that optimizes cabling efforts, by connecting the device to the nearest front-end system. It also provides savings by reducing the number of front-end computers required. Modern CPUs like the PowerPC have sufficient processing available to support a very large number of typical slow-control devices such as stepper motors, valves, viewers and magnets.

On the other hand, consolidation has inherent costs, as additional configuration management effort is required. Developers of control software for the controlled devices must concern themselves with contention for addresses, interrupts, and the availability of CPU time. Also, significant effort must go into the application management environment for the consolidated systems. There must be tools and procedures for modifying one application on the consolidated processor without affecting the other applications. The Controls Software Group at Jefferson Lab has put considerable effort into providing application developers with a framework and a set of tools that support robust low-level application management in a fashion that supports a high-availability control system [3].

The principal benefit of isolating low-level controls is its simplicity. There is no resource contention on the front-end computers, so there is less need for careful coordination among application developers. Application version management is straightforward as upgrading and rolling back the state of the systems is simplified. Independent of those attributes, isolated systems support better availability. Limiting the scope of control of front-end computers enables support personnel to address operational problems on those systems that are not absolutely required to be available during machine operation. Testing in the operational world is much simpler because the operational environment is very similar to the typical test environment, where the controls are tested in isolation. Finally, diagnosing problems on systems with fewer controls on them is more straightforward as hardware and software staff members do not have to concern themselves with interactions among a variety of control devices.

Early in the lab's usage of EPICS, when procurement dollars were limited, the low-level controls were implemented with a great deal of consolidation. Low-level RF controls were installed on one set of IOCs, because they used so much of the system resources that they would not function correctly sharing time with other applications. All other controls were grouped on other IOCs, with each IOC controlling a region of the accelerator. With the experience gained in that operating environment the lab is moving towards less consolidation. Our principal motivation is to obtain the availability improvements derived from isolated systems.

Looking forward, as the number of front-end systems increases at Jefferson Lab there will be steadily increasing support effort required just to keep the systems available. It is clear that there may be scalability issues associated with the trend towards system isolation. Before the number of front-end computers increases to 500, or even 1,000, new methods of system management will be required.

Back-end Systems

Many of the design choices made relate to segmentation and isolation of the separate operational plants, or fiefdoms, that operate at the lab. As mentioned earlier, flexibility in maintenance scheduling is vital to the lab's support efforts. The focus on maintainability and isolation of functionality has led

us to segment a number of resources according to the operational plant the resources are associated with [4]. These independent resources include separate networks, plant-specific file servers and operator consoles, and separate boot hosts for the front-end computers associated with each plant. The costs incurred with these implementation choices have been modest, principally for additional file servers and network switches. The segmentation has resulted in additional system administration effort, but that effort is minimized by keeping the control systems of the various plants as similar as possible.

Some control system resources are quite expensive, and replicating them for each plant would be a costly proposition [5]. For example, backup systems are expensive to purchase and require regular attention from system administration staff. The most cost-effective solution for the backup and recovery needs of the control system is a single backup device, robust enough to handle the 2.8 Terabytes of disk space available to the users of all the operational plants. Supporting the backup needs of the different plants requires a gigabit network connection between the various file servers and the centralized backup system, with compatible network switches between. Still, the cost of that implementation is dramatically lower than the distributed alternative, and requires much lower support effort to maintain.

Another shared resource at Jefferson Lab is the Oracle server that supports all of the operational plants. It is used for electronic logging, application management and management of operational data for each plant. Just as with the backup system, the cost of providing independent implementations of a relational database would impose large licensing costs, far beyond the budget available for this functionality. Associating the database server with a particular operational plant would present difficulties when maintenance associated with the plant results in the server being unavailable. Rather than tie the database server to one plant, then, this is implemented as a completely separate fiefdom. The result is that each plant is as independent as possible from the others, and any database-dependent functionality is available at all times, except on the rare occasions that the database server itself, or associated network, requires maintenance.

RELIABILITY

The Jefferson Lab accelerator is run more than 35 weeks each year, with round-the-clock operation when the accelerator is running. Beam delivery is the principal purpose of the accelerator, and our primary operational goal is maximization of beam delivered to the accelerator's three experimental halls. Maintenance periods lasting a day are scheduled infrequently, typically bimonthly, with semiannual extended down periods lasting several weeks. As a consequence of this schedule there is considerable effort to maximize the amount of work performed during the infrequent opportunities. This section discusses several control system design choices that facilitate such a maintenance schedule, putting a high priority on maximizing the amount of work to be performed in a short time.

The pressure to maximize beam delivery time also means that when software problems develop during operations there is considerable effort to recover quickly. The lab tracks the down time associated with problems so that manpower and procurements can be focused on the systems causing the most down time. Figure 2 shows a typical graph used to evaluate the various accelerator systems, showing the mean time between failure, and mean time to repair problems associated with the systems. The graph shows that the control system had a mean time between failure (MTBF) of 67 hours, and the mean time to repair (MTTR) those failures was 0.89 hours, or about 53 minutes.

The principal conclusion to be taken from the data as it relates to the control system is that in order to improve availability of the control system we must increase the time between failures. A repair time on the order of one hour is reasonable, because on-call personnel typically address those problems.

Maintenance

The principal reason for the segmented implementation of the control system for the different operational plants at Jefferson Lab is flexibility in maintenance. The maintenance schedule for each of the facilities is independent, but many of the resources and personnel responsible for supporting those facilities are shared. This segmentation, in addition to reducing control system costs, improves the reliability of the control system. It maximizes maintenance opportunities by enabling support personnel to take advantage of down times in a particular plant without being concerned that their work is impacting other plants.

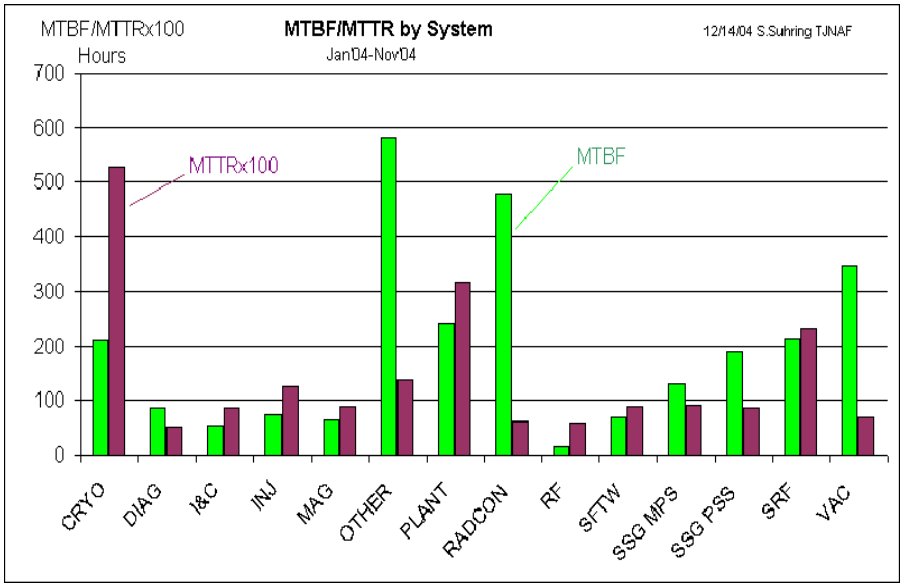

Figure 2: Mean time between failure and mean time to repair, by system

Tools

At Jefferson Lab, accelerator beam delivery time is the fundamental measure of successful operation. An automated system tracks that time, and weekly, monthly and quarterly reports are used to share the information within the lab and with the lab's primary customer, the Department of Energy. All scheduled beam time must be accounted for, with any time not used for beam delivery accounted for within the Accelerator Division's downtime reporting system. All blocks of time when beam is not delivered are attributed to subsystems associated with the problem that resulted in the downtime. With the strict accounting measures in force, the downtime data is highly reliable.

The downtime data, which has been collected for more than seven years, is used to evaluate the relative performance of each accelerator subsystem, and to determine if the subsystem is within the downtime budget allocated to it. The same data is used to track the moving average of subsystem downtime so that sudden changes in the performance can be identified and addressed. Figure 3 is an example of the six-month summary of downtime. In this example, the control system (including the categories labeled “Controls” and “Sft”) has caused the accelerator to be down for about 1% of the total scheduled beam time for the period, split roughly evenly between hardware and software sources.

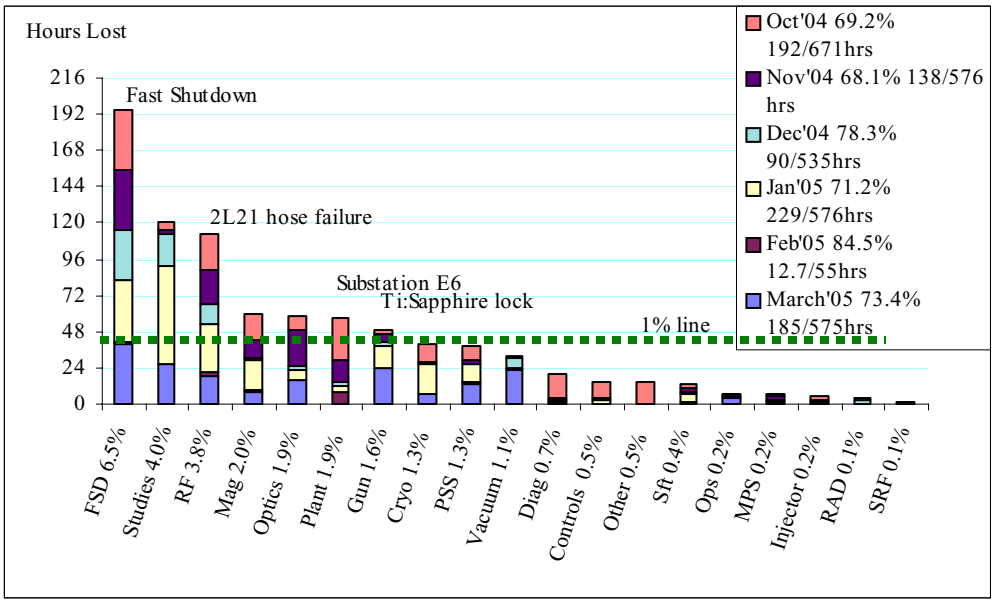

Figure 3: TJNAF 6-month time accounting by system

A detailed analysis of the last four years of downtime caused by the control system illustrates where the lab most needs to spend its effort in order to improve control system availability. While software bugs and hardware failures are a portion of the total, the majority of downtime is caused by hardware and software misconfigurations and by inadequately defined requirements. In an effort to improve the system's availability we are committing significant effort to applying more rigorous engineering principles to our development processes.

SUMMARY

This paper has discussed some of the design decisions made during the evolution of the Jefferson Lab control system. Some choices, like the implementation of a nameserver to obtain better control system performance, are apparent as the solution to observable performance problems. Solutions are less obvious when designers must trade off system maintainability against manageability. Complicating the tradeoff is the priority that system availability plays in the control system environment. At Jefferson Lab, accelerator uptime is of paramount importance. Lab management would like the accelerator to be "an electron factory", and believes it should operate as reliably as a factory. The dynamic nature of the accelerator makes this challenging, as new devices, new hardware configurations, and new experiments continuously alter the "factory floor" that must be controlled. Control system designers must respond to the reliability goals given by management while staying responsive to the dynamic accelerator environment by structuring the controls environment to support maintainability and reliability within cost constraints. At Jefferson Lab this takes the form of independent fiefdoms for each operational plant, an increasing amount of low-level controls isolation, and sophisticated software management tools that enable computer scientists to maximize their ability to take advantage of limited maintenance opportunities.

More controversial, perhaps, are the design decisions that go to the heart of the lab's drive to maximize availability. Present-day controls hardware is extremely reliable. Sophisticated analysis and debugging tools, like network- and bus-analyzers, make diagnosing low-level problems easier than ever before. Given the quality of the controls hardware and the diagnostic tools available to control system engineers, the most effective mechanisms available to enhance operational uptime are rigorous configuration management and well-designed testing procedures. Hardware redundancy and heroic efforts by control system engineers are not as cost-effective and practical at improving the performance of a complex control system as are carefully designed processes, attention to detail, and careful coordination of effort. Therefore, in parallel with its ongoing support for operations, the Jefferson Lab Controls Software Group is targeting a more mature software development process as the way to best meet the ever-more-stringent availability goals of the laboratory's operational plants.

REFERENCES

[1] J. Sage, M. Bickley and K. S. White, "Using a Nameserver to Enhance Control System Efficiency", ICALEPCS'2003, San Jose, USA, October 2001.

[2] Ding Jun, William Watson and David Bryan, "Centrally Managed Name Resolution Schemes for EPICS", ICALEPCS'1997, Beijing, China, October 1997.

[3] M. Bickley, A. Hofler, M. Keesee, S. Schaffner, D. Wetherholt, K. White "Application Management Tools", ICALEPCS'1999, Trieste, Italy, October 1999

[4] K. S. White and M. Bickley, "Control System Segmentation", PAC'2001, Chicago, USA, June 2001.

[5] M. Bickley and K. S. White, "Control System Design Philosophy for Effective Operations and Maintenance". ICALEPCS'2003, San Jose, USA, October 2001.