OLD WINE IN NEW BOTTLES—THE SPEAR CONTROL SYSTEM UPGRADE[1]

H. Rarback, C. Wermelskirchen, Stanford Synchrotron Radiation Laboratory, Stanford, CA, USA
A. Cox, Cox Realtime Corporation, Los Gatos, CA, USA

Abstract

The control systems for the SPEAR storage ring and injector were designed almost two decades ago and have worked reliably for us. Both systems are heavily dependent on the OpenVMS operating system and CAMAC. The realtime data reside in shared memory on a single computer for each control system. In order to use more modern client tools while preserving our investment in the hardware and software, we have installed an EPICS Portable Channel Access Server (CAS) on the control computers. The CAS will serve the existing realtime data as EPICS Process Variables (PVs) and allow us to use client tools like dm2k and IDL running on other workstations to more easily build new operator interfaces and develop accelerator physics programs. The CAS will also provide the infrastructure to help integrate new hardware controlled by EPICS Input/Output Controllers (IOCs).

1 INTRODUCTION

SPEAR was commissioned in the early 1970's as an electron-positron colliding beam storage ring injected by the SLAC 3-km linac. It produced some extremely important high energy physics. It is now used exclusively as an electron storage ring to produce synchrotron radiation for the Stanford Synchrotron Radiation Laboratory (SSRL) and has had its own dedicated injector for the last decade.

The original SPEAR control system was upgraded [1] in the early 1980's to a PEP I style control system. The injector control system [2] was based on a system developed at ELSA. Both systems were upgraded [3] in the mid 1990s to replace obsolete CAMAC interfaces with Ethernet based intelligent CAMAC controllers. The upgrade also replaced obsolete touchpanel display hardware on SPEAR with an X window emulation of the touchpanels.

Nevertheless, both control systems remain centralized. Although each relies critically on features of the OpenVMS operating system, they are not capable of communicating directly with each other. The SPEAR control system operator interface is intimately bound to the application programs and is primitive by today's standards. All hardware had to look like CAMAC, so that when a VME based BPM processor was installed, it had to emulate a CAMAC controller. It became clear that another upgrade was desirable.

[1] Available as SLAC-PUB-8255 at http://www.slac.stanford.edu/

2 CONTROL SYSTEM ARCHITECTURE

2.1 Before the upgrade

The control systems are of moderate size—SPEAR has about 2000 control points and the injector another 600 controlled primarily through about a dozen CAMAC crates. Figure 1 illustrates a simplified version of the system architecture. The heart of both control systems is a shared global section of memory where all the realtime control information resides. The memory is automatically backed up to disk using an OpenVMS operating system feature. Access to the data is through a different API for each of the control systems. The APIs do not allow direct network access. In addition, screens on the SPEAR control system must be programmed in a special dialect of the FORTH language,

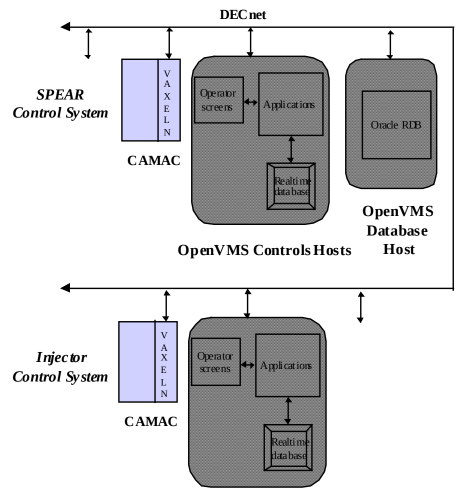

Figure 1: Original control system architecture

The application programs have been developed over two decades and represent an enormous investment in time and labor. We do not have the resources to make major changes to these applications which include the usual suite of programs necessary to maintain and operate a storage ring and injector—including machine models, BPM processing, orbit stabilization, machine protection, alarm handling, etc. Similarly, our investment in CAMAC hardware is substantial. We seek a solution to modernizing the control systems, which preserves our investment in the existing software and hardware while enabling a smooth migration path to new control system components.

2.2 *After the upgrade*

The development of the EPICS Portable Channel Access Server [4] will allow us the opportunity to use modern clients from the EPICS toolkit without major perturbations to our underlying control systems. Figure 2 illustrates how we have layered the CAS onto our two control systems. Porting the CAS C++ code was relatively straightforward because of its well thought out design. In addition, the performance of the PCAS is excellent.

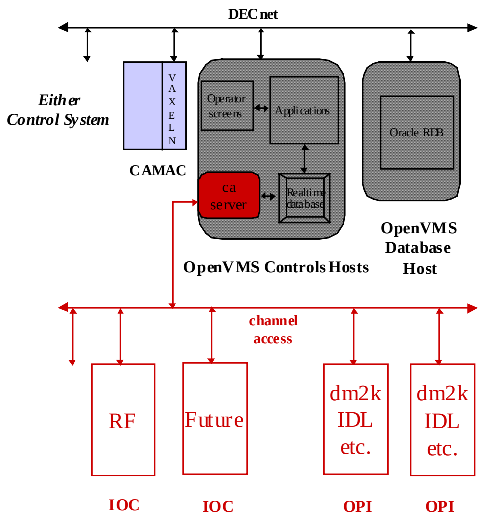

Figure 2: New control system architecture

3 THE PORTABLE CAS

3.1 *Advantages*

By using the CAS, we immediately gain access to the client tools used on EPICS Operator Interfaces (OPIs). We have already ported an EPICS editor and display manager, dm2k, to OpenVMS, but of course we are free to use dm2k and the other tools on many platforms including Windows NT and Tru64 UNIX. The freedom to use these platforms increases our flexibility to develop new applications. We can also use scripting languages like IDL, which have interfaces to channel access, to easily write maintainable applications, including sophisticated graphical applications.

These client-side tools can be used by nonexperts, in contrast to our current APIs, which are difficult to use. New interfaces for the injector and SPEAR control systems can be written to have the same look and feel. We will expect the machine operators to contribute to building new screens and machine physicists to building more generally usable accelerator applications.

As we add new hardware to the machines, especially for the upgrade to SPEAR3 [5], we will be able to interface to EPICS IOCs. In particular, the lower level instrumentation for the SPEAR3 RF system will be controlled by an IOC, and we will be installing an IOC to acquire longitudinal beam diagnostic information. Having the channel access infrastructure in place should expedite the incorporation of this new hardware.

3.2 *Implementation*

Many design decisions had to be made in order to integrate a CAS [6] with our existing control systems. The servers have slightly different designs to reflect the underlying difference in the control systems, but for the most part they share the same code. Some features of our CAS include:

- The realtime parameters are mapped to EPICS "pseudorecords". We support Analog, Binary, Multistate, and Waveform types.
- Many record fields which are required for channel access, but which did not have corresponding data in our realtime databases are maintained by the CAS in its own global section which is automatically backed up to disk. This data includes information like Alarm Severity, Access Security Group, monitor deadband, etc.
- Channel access 'get's, 'put's, and 'monitor's are all fully supported. Monitoring is done with polling in the SPEAR CAS, but uses the native Asynchronous System Traps available in the injector control system.
- The EPICS alarm system is fully supported, so that we will be able to implement the Alarm Handler.
- Most of the EPICS security model is supported. We do not support dynamic security tied to a PV, but this would not be difficult to implement.
- The servers have been optimized for performance. They use fast internal tables, which they typically only have to update the first time they see a new PV.

4 PLANS

We have just implemented the servers and so will need to gain some experience using and debugging them. One new feature we can already see a real need for is a "calc" record which runs in the server. At this point we don't have any IOCs, so that operations on PVs are not possible. If we had a calc record, which could use inputs from the local databases, we could gain a great deal of flexibility with little additional overhead.

We will be building new operator and expert screens to replace the FORTH language touchpanels on SPEAR. We also can build screens which communicate with both control systems, a feature not presently available. Complex applications like a new orbit control program will be developed in high level scripting languages like IDL or MATLAB using their interfaces to channel access.

Finally, we will be incorporating new IOCs when we need to implement new hardware. The SPEAR3 RF system and longitudinal beam monitoring will be first, but we anticipate that most new hardware will be implemented with IOCs and not in CAMAC. We will certainly investigate the possibility of running the IOCs in a non-vxWorks environment through the developing Operating System Independent layer of EPICS core.

5 ACKNOWLEDGEMENTS

We are indebted to Stephanie Allison of SLAC who has preceded [7] us in porting a CAS to an OpenVMS control system and has been very helpful in our effort.

The work reported on here was supported in part by the United States Department of Energy under contract DE-AC03-76SF00515.

REFERENCES

[1] S. Howry et al., “A Portable Database Driven Control System for SPEAR”, SLAC-PUB-3618, available from the SPIRES database at http://www.slac.stanford.edu/pubs/index.html, April 1985.
[2] C. Wermelskirchen et al., “The SSRL Injector Control System”, PAC’91, 1991
[3] R. Garrett et al. “A Control System Upgrade of the SPEAR Synchrotron and Injector”, ICALEPCS’95, October 1995.
[4] J. Hill, “Channel Access Portable Server Reference Guide”, available at http://mesa53.lanl.gov/lansce8/Epics/ca/casref/srvref-1.html
[5] R. Hettel et al., “SPEAR3, a Brighter Source at SSRL”, PAC’97, May 1997.
[6] A. Cox, “Program Notes for the Injector and SPEAR Portable Channel Access Server”, available from the authors, September 1999
[7] S. Allison, “Introduction of EPICS into the SLC Control System”, http://www.slac.stanford.edu/grp/cd/soft/ipanel/html/indx98.html., October 1995.