The Duke Storage Ring Control System *

Y. Wu, B. Burnham, and V.N. Litvinenko
Duke FEL laboratory, Box 90319, Duke University, Durham, NC 27708-0319, USA

Abstract

The Duke storage ring is a dedicated facility for the UV-VUV FEL operation. The low level computer control system for the Duke storage ring is developed using EPICS. The control hardware employs several different architectures including CAMAC, GPIB, Allen Bradley, and VME. The high level control is implemented in Tcl-Tk scripts running on SPARCstations. Tcl-Tk provides the global control capabilities such as the energy ramping, the orbit compensation, and the tune and chromaticity control. The Duke storage ring control system was tested and operational for storage ring commissioning in Nov. 1994. During commissioning, additional control tools were developed to facilitate the operation.

I. INTRODUCTION

The Duke storage ring control system is based on the Experimental Physics and Industrial Control System (EPICS) [1]. EPICS was initially co-developed by Los Alamos National Laboratory and the Advanced Photon Source. Later, the co-development was joined by Lawrence Berkeley Laboratory (LBL), the Continuous Electron Beam Facility (CEBAF), and Deutsches Elektronen-Synchrotron (DESY).

EPICS is a tool-based control system. EPICS tools include the display manager, the alarm manager, the archiver, the sequencer, the channel access, and others. These tools are software modules with carefully defined layers and boundaries. The tool-based approach enables EPICS implementations to use completely different sets of tools, and allows independent development of EPICS at different sites.

The EPICS control system is also a distributed system comprised of operator interfaces (OPIs), input output controllers (IOCs) and local area network (LAN). Using EPICS, a sophisticated control system customized for a particular accelerator can be quickly implemented, tested and ready for operation, saving a tremendous amount of time and effort to develop a home-made control system.

The development of the EPICS control systems at Duke FEL lab is one of many success stories of the EPICS control system. Duke FEL lab was one of the early EPICS users. The first EPICS control system at Duke, the Mark III FEL control system, was completed and operational in September, 1993, under the direction of Dr. Ricardo Pantazis. In March, 1994, we started the major effort in developing the EPICS control system for the storage ring. On November 1, 1994, the basic functions of the control system were ready for storage ring commissioning.

*Work supported by U.S. Air Force Office of Scientific Research Grant F49620-93-1-0590 and U.S. Army Space & Strategic Defense Command Contract DASG60-89-C-0028.

Additional information about EPICS is found in [1], [2],[3], and their references. More information about the Duke storage ring commissioning, is found in [4].

II. REQUIREMENTS

The requirements of the Duke storage ring control system include the following:

- control for various hardware systems;
- functional control of a group of channels;
- control of beam instrumentation and diagnostic systems;
- machine modeling and intelligent controls.

In the first phase of the control system development, we concentrated on the hardware control and the functional control. Using EPICS we implemented controls on hardware systems, such as the magnet system, the RF system, the injection kicker, the vacuum system, the water cooling system, and diagnostic systems. Using Tcl-Tk we developed high level control tools to provide functional controls such as the energy ramping, the orbit compensation, and the tune and chromaticity control.

In addition, we implemented the EPICS alarm handler, archiver, and sequencer for the storage ring control system.

III. LOW LEVEL SOFTWARE DEVELOPMENT

Since serial CAMAC device-drivers were not available from EPICS when we began the control system development in March, 1994, we had to write our own device-drivers for serial CAMAC using the CEBAF low level CAMAC C library (developed by Marty Wise).

We have developed a standard driver for the serial CAMAC branch driver Hytec2992 to initialize CAMAC crates and to provide a uniform way for device codes to address CAMAC. We have also developed a set of device codes for DSP3016 (16-bit DAC), DSP2032 or BiRa5305 (16-bit ADC, SAM module), BiRa 2324/3224 (24-bit binary input and output modules), and Russian 20-bit ADC. In addition, we have developed a driver code for the GPIB-CAMAC interface board KS3388 and modified the standard EPICS driver for the Analogic DVX2502 to suit our application.

IV. HARDWARE SUBSYSTEMS

The Duke storage ring control system employs a mixed architectures of CAMAC, Allen Bradley, GPIB, and VME. The physical layout for the control system is shown in Fig.1.

The main control functionalities are implemented in CAMAC, because we have a large amount of CAMAC control hardware on hand. The CAMAC system includes the controls for dipoles, quadrupoles, and correctors, the readbacks for the dipoles, correctors, and the beam current monitor. Allen Bradley analog and binary modules are used to control the relatively slow subsystems, such as the water cooling, the vacuum, and the RF

cavity. The GPIB-CAMAC interface provides the communication between the CAMAC controller and the HP3458A multimeter. The HP3458A is used to precisely measure the quadrupole currents. The Analogic DVX 2502 board is used to read the quadrupole terminal voltages which are in turn used together with the quadrupole currents to calculate the quadrupole temperatures.

Overall, the present storage ring control system includes 278 analog inputs, 265 analog outputs, 48 binary inputs, and 23 binary outputs.

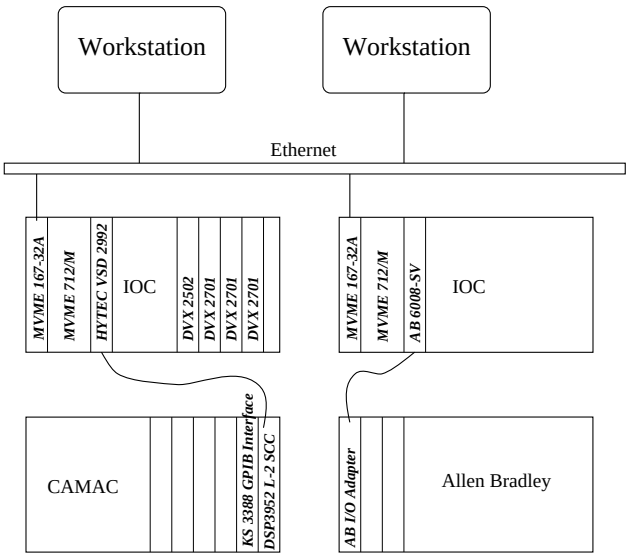

Figure. 1. Hardware layout for the Duke storage ring control system.

V. IMPLEMENTATION OF EPICS

We implemented the storage ring control system based on EPICS release 3.11. The main effort has been devoted to developing and implementing the control database, the operator interface, the alarm handler, and the archiver.

Besides the regular analog and binary records, the control database contains a relatively large number of subroutine records. These records are used to accomplish complex control functionalities. One example is the magnet control. To utilize data from the magnetic measurements, we have analyzed the measured data and fit them to splines or high order polynomials which are later used in the subroutine records for the precision control of magnets (see [5]).

The storage ring operator interface is developed using the **edd/dm** tool. **edd/dm** has provided us an easy way to control and monitor the system. The alarm handler **ahl** is used to alarm abnormal quadrupole temperatures. We also have implemented the archiver, **ar**, to log the vacuum and beam current data for analysis.

VI. HIGH LEVEL CONTROL

The control of a physics application such as a storage ring requires a flexible means of handling the complex functional control of a group of hardware channels. However, when a change is made in the EPICS device driver, database, or subroutine, an IOC reboot is needed to update the change. This approach limits the ability to construct new functional controls at run time, and diverts the effort of the application software development to the low level control debugging and implementation.

To avoid these problems, the scripting language – Tcl-Tk [6] can be used to build a dynamic, flexible and application oriented high level control.

A. Tcl-Tk

Tcl and Tk are software packages developed by Dr. John Ousterhout. Tcl (tool command language) is a scripting language for controlling and extending applications. Tk is an X window toolkit extension to Tcl, providing commands for building graphic user interfaces. Tcl is an *embeddable* language, which provides an easy way to incorporation other C procedures to extend the core Tcl features. Besides Tk, a number of other Tcl extensions have been developed including an interface to the EPICS control system – Tcl-CA (developed by Johannes van Zeijts at CEBAF).

Because of the nature of the scripting language, Tcl programs can be written and used interactively without compiling. This enables us to develop, test and debug control software on fly, saving a lot of time in code debugging, compiling and loading.

We have developed the Duke storage ring high level control using Tcl-Tk. An executable Tcl program with extensions such as Tcl-CA, TclX (an extended Tcl) and BLT (a graphic tool extension to Tk) is compiled and renamed as **epicswish**.

B. High Level Control

The high level control is made up of several Tcl scripts, including a control script, a kernel script, and a number of application scripts. The control script serves as a centralized control for the application loading and initialization. The kernel script, comprised of six procedures, provides basic communications between IOCs, workstations, and data files. The core of the functional control is implemented in various application scripts. The application scripts are summarized here:

1. a script to ramp the storage ring energy;
2. a script to control storage ring magnets and correctors — useful for magnet or corrector normalization;
3. a script to control dipole trim coils for closed orbit compensation;
4. a script to change tunes and chromaticities;
5. a script to take snapshot of or restore the control system configuration;
6. a script to modify a configuration file for use at a different energy and/or at a modified lattice configuration;
7. several Tk scripts to provide graphic interfaces to other application scripts.

A block digram, Fig.2, summarizes the Duke storage ring high level control.

As an example, we illustrate the storage ring operation procedures in Fig.2. In an interactive **epicswish** window, we first load and initialize all related scripts by sourcing *control.tcl*. To prepare for injection, we call a *mod* procedure to modify an old snapshot *snp_03_01_95*. An editor is provided by *mod* to specify the injection beam energy, and changes in tunes, chromaticities,

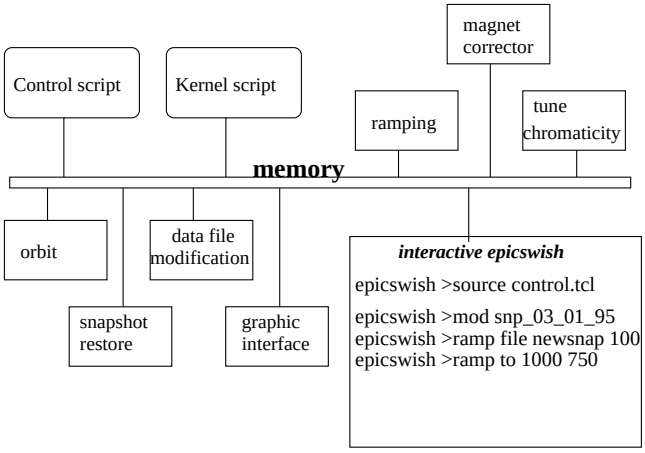

Figure. 2. Block diagram for the Duke storage ring high level control implemented in Tcl-Tk.

and/or the setting of a group of channels. Using standard lattice data files and interpolation methods, a new data file, named *newsnap*, is created. After ramping the storage ring to the new configuration (*newsnap*) in 100 steps, the ring is ready for injection. Finally, we ramp the stored beam current to 1000 MeV using standard configuration files in 750 steps.

As in this example, we successfully ramped the electron beam energy from an injection energy of 230–250 MeV to the designed 1100 MeV at the beginning of the storage ring commissioning [4].

VII. PERFORMANCE

The Duke storage ring control system consists of 614 I/O points and 1423 EPICS database records. The database records are distributed in two IOCs — "ring" and "ringmag". The "ring" IOC consisting of 195 records is lightly loaded, while "ringmag" IOC is heavily loaded with 1228 records, including 155 calculation records and 444 subroutine records. Since the beginning of commissioning in November, 1994, the storage ring control system has been practically trouble free, with only one IOC lockup due to a hung channel access process in the IOC.

The Tcl-Tk high level control has provided us with global functional control with reasonable performance. For example, the ramping script is capable of updating 240 control channels in about 0.5 second. Using this script, we are able to ramp electron beam from the injection energy of about 250 MeV to the design energy of 1000 MeV in about 6 minutes at 1 MeV per step.

VIII. CONCLUSION AND FUTURE DEVELOPMENT

Overall, the Duke storage ring control system is a stable, reliable, and easy to operate system. Using Tcl-Tk, we are able to continuously develop and improve high level control tools to meet the demands of the storage ring commissioning. With the combination of EPICS and Tcl-Tk, we were able to develop a sophisticated control system for the Duke storage ring within eight months with only two developers.

In the future, we intend to develop a fast ramping system by implementing a ramping mechanism in IOC. We also intend to improve the performance of the high level control by replacing time consuming Tcl programs with C routines. At the same time, we will provide additional graphic interfaces to simplify storage ring operation.

In addition, the second phase of the control system development will focus on the development of beam instrumentation and diagnostic systems, as well as machine modeling and intelligent control.

IX. ACKNOWLEDGEMENTS

We are very thankful to Dr. Ricardo Pantazis for his work on the EPICS system implementation at Duke, as well as his early work in the storage ring control development. We would also like to thank Carl Dickey, Henry Goehring, Jim Meyer, and Owen Oakeley at Duke for their help in the control hardware implementation. We are very grateful to the EPICS community from which we obtained not only the EPICS control software, but also the constant support through its listservice on the net. We are especially thankful to the people at CEBAF: Chip Watson, Marty Wise, Johnny Tang, and Johannes van Zeijts for providing us with their low level CAMAC C library, CAMAC array input and output records, and Tcl-CA.

References

[1] L. R. Dalesio, M. R. Kraimer, and A. Kozubal, "EPICS Architecture," Proceedings of ICALEPCS, KEK, Tsukuba, Japan, 1991, pp.278–282.

[2] L. Dalesio, *et al.*, "The Experimental Physics and Industrial Control System Architecture," Proceedings of ICALEPICS, Berlin, Germany, Oct. 18–22, 1993.

[3] W. McDowell, M. Knott, and M. Kraimer, "Status of the Advanced Photon Source and Its Accelerator Control System," ICALEPICS, Berlin, Germany, Oct. 18–22, 1993.

[4] V. N. Litvinenko, *et al.*, "Commissioning of the Duke Storage Ring," these proceedings.

[5] B. Burnham, *et al.*, "Application of Precision Magnetic Measurements for Control of the Duke Storage Ring," these proceedings.

[6] J. K. Ousterhout, "Tcl and the Tk Toolkit," Addison-Wesley Publishing Co., Massachusetts, USA, 1994.