Startup of the Experimental Physics Industrial Control System at NSTX*

P. Sichta, J. Dong
Princeton Plasma Physics Laboratory, P.O. Box 450, Princeton, New Jersey 08543

Abstract--The Experimental Physics Industrial Control System (EPICS) is a set of software which is being used as the basis of the National Spherical Torus Experiment's (NSTX) Process Control System, a major element of the NSTX's Central Instrumentation & Control System. EPICS is a result of a co-development effort started by several U.S. Department of Energy National Laboratories. EPICS is actively supported through an international collaboration made up of government and industrial users. EPICS' good points include portability, scalability, and extensibility. A drawback for small experiments is that a wide range of software skills are necessary to get the software tools running for the process engineers. Our experience in designing, developing, operating, and maintaining NSTX's EPICS (system) will be reviewed.

I. Why EPICS?

During the NSTX design phase the group responsible for the central control system assimilated system requirements. Evaluation criteria was developed and a product search was conducted to elicit candidate systems. The selection process eventually led to the choice of EPICS [1,2] as the software for the NSTX Central Instrumentation and Control System [3] (CI&C). There was an optimistic undertone that once NSTX had some experience with EPICS, it might be a catalyst in providing benefits to the national fusion program, as observed in the accelerator and telescope communities.

The EPICS software was available at no cost to the NSTX project. Much of the non-EPICS auxiliary software was freely available from the "GNU Project" and other sources. This was important in order to remain within the extremely tight NSTX construction budget. Of course, the control system wasn't free, operating systems and hardware had to be selected and acquired.

Technically, EPICS was attractive because it is:

- completely open,
- extensible,
- capable of running on Unix,
- runs on computers from competing suppliers,
- users were already using similar types of process I/O hardware,
- scalable, to support future growth.

The only factor meriting special consideration was technical risk. While commercial alternatives can offer the security of product support and training, experience has shown that attention to small-market products does not always match the urgency of the customer. In other words, given the open nature of EPICS coupled with the listserver-based communications among users, problem-resolution can be put under control of the project.

II. Getting EPICS Running

EPICS is a suite of software applications running in a client-server model. Fig. 1 shows a hardware block diagram of an EPICS system.

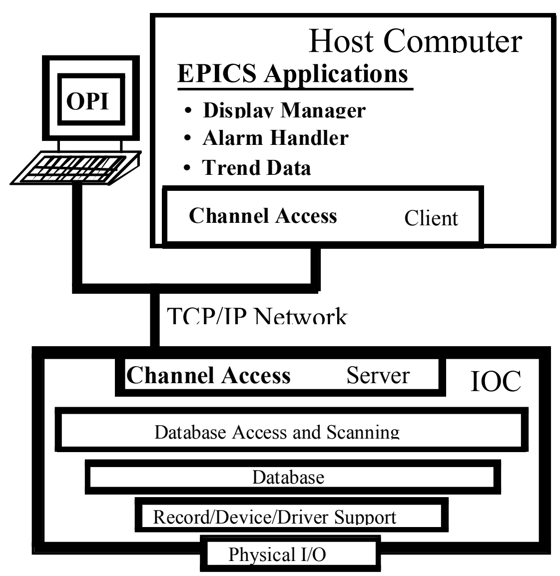

Fig. 1 Basic EPICS Architecture

* Work performed under the auspices of the USDOE by Princeton Plasma Physics Laboratory under Contract No. DE-AC02-76CH03073.

The server software runs on the input-output controller (IOC), a VME crate with a single-board computer running the vxWorks real-time operating system. The IOC's contain the logic and sequencing for controlling the process, and hardware modules for device-level I/O. The client software runs on the "host" computer, a Sun workstation running Solaris. Operators use an X-window display and the client programs to command the process-control software running in the IOC. An EPICS system can have any number of IOC's, hosts, and X-terminals.

Using EPICS is easy. Getting EPICS to run can be difficult for inexperienced users. Unlike most modern commercial software packages you cannot merely pop in a CD-ROM and hit the "Install" button. Software and licenses must be purchased from several vendors. Several EPICS, GNU Project, and other software utilities must be downloaded from various sites on the internet. Once the utilities are downloaded, they must be placed in proper directories, environmental variables set, paths defined, privileges granted, etc… and then compiled using the unix Make utility.

Once the build-tools have been installed the building of EPICS base can begin. This was not a trivial process even though there were installation guides available. However, implementing the simply-stated steps in the guides took several weeks. The personnel developing the EPICS at NSTX were hardware and process-control oriented and not very experienced at unix administration. Fortunately, others at the lab who had such experience were available for consultation.

III. Wide Range of Skills

A wide range of skills are required to support an EPICS system. This can present a challenge to small EPICS development groups. The aforementioned unix system administrator is needed first. Then programming skills are typically used to configure and compile EPICS extensions and to write or debug record, device, or driver routines. Hardware engineering skills are required to configure the IOC and I/O modules. Connecting a PLC may require that the communication protocol and other interfacing issues need to be understood in detail. And a well-designed network is critical to achieve good system performance.

A lack of practical experience in some of these areas can lead to delays and frustration. If there is a problem a small group may not have the expertise to quickly identify and correct the cause. For example, the NSTX startup effort lost several weeks due to faulty memory on the first IOC computer that was installed. The bad hardware was not discovered for several weeks, partly because of confidence in the (new) Motorola hardware, and a commensurate lack of confidence in the manner in which the (new) software was being used.

Table 1 summarizes the major skills needed to get the EPICS base software to run. Each skill has a rating between one and three. A three indicates that proficiency is essential for a fast development cycle.

Table I. Skill/Proficiency for EPICS Building

Skill	Rating
Unix administration	3
C programming (C++ use growing)	3
vxWorks programming	2
network design	2
Make/GNUmake knowledge	2
Tornado/vxWorks administration	1
network administration	1
X-manager administration	1
firewall administration	1
VME and single-board computers	1

Fig. 2 below shows the relative sequence of when the various skills were required. The operating system administration effort must occur first, and eventually the process engineer (user) becomes involved when defining the process-control algorithms. Once the EPICS base has been established the system administrator and software engineer assume a much smaller support role.

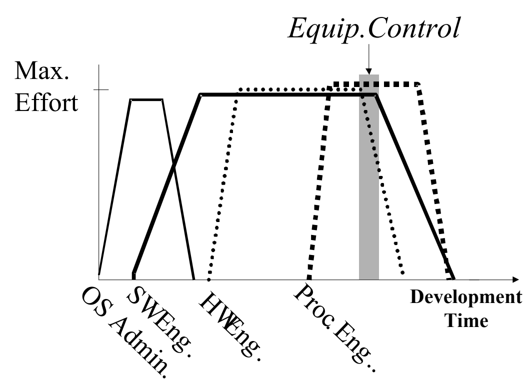

Fig. 2 Diagram showing the sequence of skills necessary to bring EPICS to a working state. Skills shown are operating system administration, software engineering, hardware engineering, and process-control engineering.

IV. Development Issues

The EPICS software was successfully used in achieving the NSTX project's initial milestone, first plasma. A retrospective of our EPICS development experiences show that a few precautionary measures during the early development phase could have increased the control system's safety margin for operational readiness.

A. Late Start

It would have been beneficial to begin using EPICS sooner. The NSTX construction schedule had the CI&C effort as one of the last to begin in earnest (i.e. labor funding ramp-up). The very first exposure to EPICS was at a (worthwhile) 4-day EPICS Training course at Los Alamos. This was less than one year prior to first plasma. After the course was completed, it was still several months before an on-site, working version of EPICS was available for beginning the hands-on learning curve.

B. No Prototype

Not building a small, low-performance prototype system early on was another problem. A prototype could probably have been constructed using "spare" parts. To constrain costs, working on the prototype could have been approached as a low level of effort task, spanning months. This should have been done well ahead of the time when the project's controls wheel needed to be in high gear. The importance of this cannot be overstressed. The most beneficial aspect of a prototype would be to provide a realization of the skills needed to get EPICS built.

C. Get the Right People

It took much longer than anticipated to create the software environment for building EPICS. Access to an experienced unix system administrator was essential for building and installing EPICS.

D. GDCT

To increase productivity a graphical configuration tool was used to produce the record databases that are loaded into the IOC. The editor of choice was called GDCT, which was open source and part of the EPICS distribution. There was an alternative to GDCT, it was based on a commercial product called Capfast. That was not selected because of cost. GDCT is no longer being maintained, and whatever bugs are in it will remain there. However, once recognized, the bugs can usually be avoided. NSTX did not have the resources to fix the product for the collaboration. To improve software maintenance, it would be beneficial to begin using a better database tool.

V. Conclusion

It has been over a year since EPICS has been running at NSTX. The system has been very reliable and new applications can now be quickly added. One remaining issue is in the combining of records to implement a new control algorithm. It is not always intuitive which record fields should be to linked together to get the desired result. However, with a little trial and error the algorithm is soon working as intended.

Control system operations are now routine. Enhancements, reliability, and maintainability issues have become the focus of the control group. There are several additional EPICS tools that can be brought online to enhance the usefulness of the system. Keeping our system current with new EPICS releases (about every 12 months) and Solaris releases are important maintenance tasks.

References

[1] EPICS documents at Argonne National Laboratory, Argonne, IL, *available over the internet.*

[2] EPICS documents at Los Alamos National Laboratory, Los Alamos, NM, *available over the internet.*

[3] P. Sichta, et al , "The NSTX Central Instrumentation & Control System ," *18th Symposium on Fusion Engineering*, Albuquerque, NM (1999).