EPICS Server-Level API Developers Survey

J. O. Hill, LANL, USA

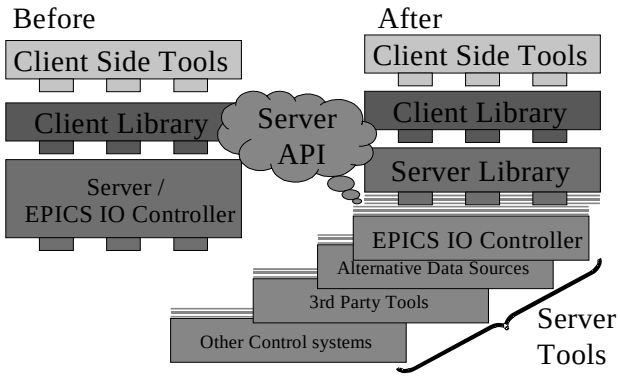
Abstract

The Experimental Physics and Industrial Control System (EPICS) [1,2,3,4,5,6] collaboration has more than 100 members worldwide. This collaboration has a joint open-source software development initiative to produce modular control system software components. A new network server side application programmer's interface (API) for EPICS described earlier[1] was expected to increase the utility and flexibility of the EPICS software. We briefly describe some of the many applications that are now using this API, and how it *has* led to new uses for many components in the EPICS software distribution.

1 INTRODUCTION

In the past the EPICS server and the EPICS function-block-based process control system were inseparable. After inserting a well-defined API between these components it was hoped that EPICS could be used with a variety of plug-compatible data sources that we have labeled as "server tools" (Figure 1).

Figure 1: Server Level API Adapts EPICS to New Applications



A previous paper [7] envisioned that certain situations would be appropriate for creation of a server tool: an alternative data source leverages the mature EPICS client side tools[2] (interoperability), a client computed result is needed by other clients (modularity), or clients need to access critical servers via a proxy server (scalability). It was also hoped that this new API would promote sharing of components between sites, facilitate integration between dissimilar systems, and require only a modest amount of effort on the part of the server tool developer. In October of 1998, with its objective to determine the validity of the previous paper's intuition, a survey was sent out using the EPICS Internet mail list. Server tool developers were asked the following questions: are you using the EPICS server library; what is your application; and what is your overall impression of the software so far? This paper will first look at the depth of the server tool applications responding, attempt to summarize the comments in the survey responses received, and it will briefly summarize our reaction to the issues raised by the survey respondents.

2 SERVER TOOL APPLICATIONS

The developers were asked about the nature of the server tool applications that they were designing. There were 18 developers responded to the survey, and among these, 22 independent server tool applications were developed (Table 1). These server tools were running on 8 different operating systems: Sun-Solaris, HP-UX, LINUX, SGI-IRIX, Microsoft-WIN32, DEC-VMS, Apple-MAC, and WRS-vxWorks. From the range of applications in the response, and the diversity of operating systems employed, it can be safely concluded that the software is general purpose and portable. Ports to new platforms were performed by some of the survey respondents, and not always by the library developers. This appears to confirm that the server library software is properly organized.

Table 1: Server Tool Applications

Developer's Site	Server Tool Developed
APS-BESSRC-CAT	TCL / TK interface (planned)
APS-SRI-CAT	NT based Digital Camera
APS-SRI-CAT	Motion Control
APS-ASD	EPICS Proxy (Concentrator)
APS-IMCA-CAT, APS-MR-CAT	MX Data Acquisition Toolkit Gateway (development in progress)
PSI	Video (planned)
ORNL	Low Cost Serial IO
MIT-BATES	Facility Control System Gateway (planned)
SLAC-SPEAR	SLAC-SPEAR Control System Gateway
SLAC-PEPII	SLAC-PEPII Control System Gateway
BaBar Detector	Data Acquisition System Gateway
DESY	DOOCS Control System Gateway
KEKB	LINAC Control System Gateway
KECK Telescope	Command Processor Subsystem Gateway
KECK Telescope	Telescope Simulator
LANL-LEDA	IDL (4th Generation Language)
LANL-LEDA	Active X conversant programs such as LabView
LANL-LEDA	NT based Digital Camera
LANL-LEDA	Directory Service
LANL-LEDA	RF Fault Log
MSU-NSCL	NSCL Control System Gateway
MSU-NSCL	Modicon PLC Gateway

3 IMPRESSIONS OF THE SOFTWARE

A limited set of the respondents provided feedback on their overall impression of the software. Their comments have been included condensed, but otherwise unabridged, in order to better support the accuracy of the summary conclusions below. Table 2 is a complete list of comments on the software. Table 3 contains all of the comments received concerning the documentation. After closely examining these comments some conclusions can be drawn about the majority opinions of the developers. The biggest weakness of the software appears to be the complexity and efficiency of our data descriptor object based mechanism for passing data. Nevertheless, while some of the comments indicated that developers perceive that certain aspects of the API are unnecessarily complex, there also appears to be a general consensus that a server tool can be created within a modest expenditure of effort. From the responses here, and also the lack of negative responses, it appears that there is general agreement among the server tool developers that the software is useful and reliable. From the survey responses we also conclude that there are improvements that need to be made to the documentation.

Table 2: Impressions of the Software

Developer's Site	Comments Returned
APS-BESSRC-CAT	The run time type conversions possible in the data descriptor library are rarely necessary. The deletion policy for server objects is confusing. A simple, lightweight, and possibly C++ template based, process variable class implementation is desired.
APS-SRI-CAT	The interface is a bit complex, but I am not a C++ expert. The server API leverages the large amount of software written for the PC into EPICS. The library has been reliable.
APS-IMCA-CAT, APS-MR-CAT	The software is a reasonably straightforward and understandable package.
ORNL	An excellent resource that has provided a high quality general-purpose solution for a few man-weeks of effort.
SLAC - PEPII	Our server tool was easy to implement, but the data descriptor library was unnecessarily complex.
BaBar Detector	Our experience has been pretty smooth, but difficulties occurred when using the string class in the data descriptor library.
KECK Telescope	The software was generally reliable and efficient, but problems occurred with large arrays* and when asynchronous IO completed after the initiating client disconnected* * Author's note: patches for both of these problems have been installed into the distribution
MSU-NSCL	An inexpensive and low effort alternative.

Table 3: Impressions of the Documentation

Developer's Site	Comments Returned
APS-BESSRC-CAT	very helpful
MSU-NSCL	the source code embedded in the documentation didn't compile.
SLAC-PEPII	good server library documentation
NMSU	nothing that we don't understand so far
KECK Telescope	more documentation desired, but LANL web documentation has not yet been consulted

4 OUR REACTION TO THE SURVEY

Concise, efficient, and backwards compatible API alternatives to the current C++ data descriptor object based mechanisms for passing data are under investigation. A C++ abstract base class is the current leading candidate for this role. Documentation upgrades are also mandated.

5 CONCLUSIONS

The depth of applications in the response indicates that the software is general purpose and portable. From the comments returned we conclude that developers perceive that data description aspects of the API are unnecessarily complex, but nevertheless, a majority of the developers agree that server tools can be developed within a modest amount of effort.

As envisioned during ICALEPCS 95, the EPICS server level API has facilitated increased sharing of software components between sites, increased our freedom to choose and combine components of EPICS, and made it possible to integrate EPICS with a wide range of 3rd party tools and dissimilar systems. This has transformed EPICS beyond its distributed process control system roots, and prepared it for supporting roles integrating the wide range of dissimilar systems omnipresent in complex projects.

REFERENCES

[1] M. Knott et al, ''EPICS: A Control System Software Co-development Success Story", ICALEPCS'93, Berlin, 1993.

[2] W. McDowell et al., ``EPICS Home Page", ``http://www.aps.anl.gov/asd/controls/epics/EpicsDocumentation/WWWPages/EpicsFrames.html".

[3] L. Dalesio et al. ``The Experimental Physics and Industrial Control System Architecture: Past, Present, and Future", ICALEPCS'93, Berlin, 1993.

[4] L. Dalesio et al, ``The Los Alamos Accelerator Control System Database: A Generic Instrumentation Interface", ICALEPCS'89, Vancouver, 1989.

[5] J. Hill, ``Channel Access: A Software Bus for the LAACS", ICALEPCS'89, Vancouver, 1989.

[6] J. Hill, ``EPICS Communication Loss Management", ICALEPCS'93, Berlin, 1993, pp 218-220.

[7] J. Hill, "A Server Level API for EPICS", ICALEPCS'99, Chicago, Oct 1995.