MIGRATING TO AN EPICS BASED INSTRUMENT CONTROL SYSTEM AT THE ISIS SPALLATION NEUTRON SOURCE

M. J. Clarke, F. A. Akeroyd, K. Baker, G. Howells, D. Keymer, K. J. Knowles,
C. Moreton-Smith, ISIS, STFC Rutherford Appleton Laboratory, Oxon, UK
Kevin Woods, Tessella, Abingdon, Oxon, UK

Abstract

The beamline instruments at the ISIS spallation neutron source [1] have been running successfully for many years using an in-house developed control system. The advent of new instruments and the desire for more complex experiments has led to a project being created to determine how best to meet these challenges. Though it would be possible to enhance the existing system, migrating to an EPICS-based [2] system offers many advantages in terms of flexibility, software reuse and the potential for collaboration. While EPICS is well established for accelerator and synchrotron beamline control, is it not currently widely used for neutron instruments, but this is changing. The new control system is being initially developed to run in parallel with the existing system, a first version being scheduled for testing on two newly constructed instruments starting summer 2013. In this paper, we will discuss the design and implementation of the new control system, including how our existing National Instruments LabVIEW controlled equipment was integrated, and issues that we encountered during the migration process.

BACKGROUND

The ISIS spallation neutron source has been providing world-class science since the mid-1980s and currently has over thirty beamline instruments producing world-leading research. The software and hardware used to control these beamline instruments plays a significant part in the productivity of ISIS.

The original instrument control system was run on the OpenVMS [3] platform using in-house developed software. Around the year 2000, the control system started to move away from using the OpenVMS platform to using PCs running Microsoft Windows NT [4] partly because of concerns over the longevity of OpenVMS and partly because software vendors were starting to give priority to Windows. As part of this move, it was necessary to replace the existing control system software. Since vendors were increasingly supplying LabVIEW controlled equipment, LabVIEW was chosen to form the basis of the new system. The current control system has evolved from this and now consists of LabVIEW drivers for equipment control alongside in-house developed software for scripting, neutron data collection and experiment management, etc.

The more recently constructed beamline instruments are capable of performing significantly more complex experiments than older instruments and, as a result, are starting to push beyond the boundaries for which the existing control system was designed. With this in mind, and with the prospect of even more complex instruments being built in the future, it was necessary to address the future direction of the instrument control system.

THE EXISTING CONTROL SYSTEM

The existing control system runs on a Windows 7 x64 Virtual Machine (VM). The key advantage of using a VM is that it adds a layer of abstraction between the hardware and the operating system; this means that the VM can easily be transferred to another host PC if there is a hardware issue.

Drivers for beamline equipment are written using LabVIEW. LabVIEW makes it relatively easy to create a driver with a built-in Graphical User Interface (GUI) known as a Virtual Instrument (VI). The vast majority of VIs used at ISIS are written in-house.

The neutron data is accumulated in the Data Acquisition Electronics (DAE); the DAE is controlled using an in-house developed service called the Instrument Control Program (ICP) which is written in C++/Microsoft ATL.

The control system is managed by SECI which is essentially a configuration and windows manager. SECI is responsible for loading and managing the VIs, providing an instrument status summary and logging beamline data. As SECI is the main GUI of the system, it also shows a summary of the critical data from the other parts of the system. The layout of the existing control system is illustrated in Fig. 1.

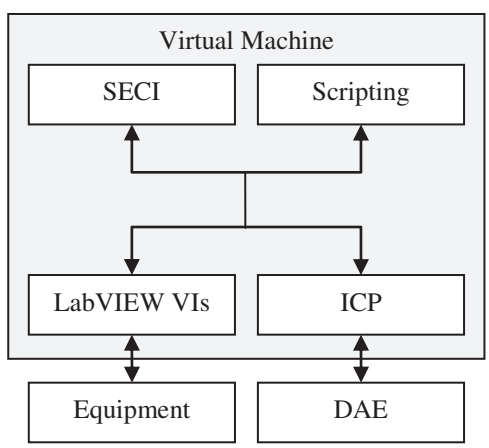

Figure 1: The layout of the existing control system.

ISBN 978-3-95450-139-7

Scripting is provided by a bespoke command line interpreter called Open GENIE [5]. Open GENIE allows users to control the whole system from the command line and automate a series of experiments via scripts.

Microsoft's DCOM protocol is used to communicate between the different software components that combine to form the control system.

WHY EPICS?

A review of the existing control system identified a number of potential weaknesses, with the key ones being:

- Strong coupling between different components in the system
- No defined boundaries between drivers, business logic and Graphical User Interfaces
- Lack of opportunities for collaboration or to make use of code shared by other facilities

The combination of strong coupling and lack of defined boundaries means that replacing or upgrading components is difficult and can lead to unforeseen problems in other areas. This also means that it is difficult to adapt or extend the system to meet future needs.

The computing group at ISIS is quite small, so the opportunity to collaborate with other institutions or use existing code would be advantageous.

It was decided that building a new control system around EPICS would allow the identified areas of weakness in the existing control system to be addressed. EPICS, by its nature, would allow the creation of a loosely coupled system with defined boundaries between the drivers and other layers.

There are other similar systems that could have been chosen, but an important factor in selecting EPICS is the strength of the community, especially as both the Diamond Light Source (UK) [6] and the Spallation Neutron Source (USA) [7] are active members. Diamond Light Source have significant EPICS expertise and being located on the same campus as ISIS makes it very convenient. Like ISIS, the Spallation Neutron Source is also in the process of changing their beamline instruments to use EPICS; this provides opportunities for collaboration and knowledge sharing.

THE PROJECT

Prior to the start of the development phase of the project, an extensive consultation and requirements gathering exercise was performed. Members of the ISIS computing group met with the scientists responsible for the beamline instruments to discuss strengths and weaknesses of the existing control system, and features required from the new system. The initial project requirements document was produced from this consultation exercise.

The development phase of the new control system started in December 2012. Two under construction beamline instruments (CHIPIR and LARMOR) were identified as appropriate candidates to trial the first version of the new EPICS-based system. The CHIPIR instrument was scheduled to be in a commissionable state from July 2013 onwards, with LARMOR being ready by September 2013. This meant there was approximately seven months available to develop the first version of the new control system from the project commencement.

The Scrum framework [8] was chosen as the methodology for the development process. Scrum is an agile software development framework; the main features of which are:

- The development period is divided up in to smaller periods called "sprints" (typically one to six weeks long)
- The project requirements are stored in the "backlog" with a priority determined by the customer
- At the beginning of each sprint, a sprint's worth of work is chosen from the backlog
- There is a daily stand-up meeting where each team member must state what they have been doing, what they are going to do, and if there are any impediments to progress
- At the end of each sprint there is a demonstration for the customer of a usable piece of software
- At the end of each sprint the backlog is refined to take into account any new requirements or changes in priority

The key advantages of an agile development process, like Scrum, are: the customer is involved throughout; requirements can be easily changed at any point in the process; and, project progress can easily be monitored.

THE STRATEGY

To minimise the risk of commissioning/experiment time being lost in the event of a problem with the new control system, it was agreed that the first version of the new control system would be developed to run in parallel to the existing system. To facilitate this, it was decided that the EPICS-based system would run on a separate Virtual Machine to the existing system but connect to the existing LabVIEW VIs and the ICP as illustrated in Fig. 2.

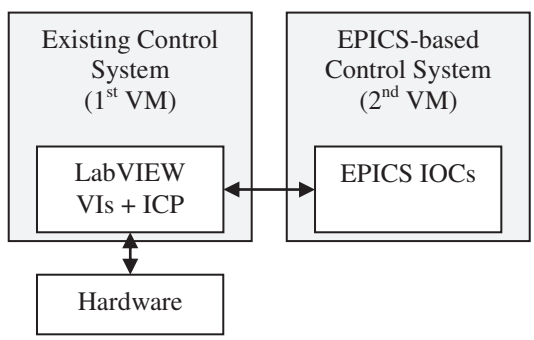

Figure 2: The parallel control system.

The advantage of initially incorporating existing VIs into the EPICS-based system is that it allows early development effort to be applied to other critical parts of

ISBN 978-3-95450-139-7

the system, such as configuration, deployment, user interfaces and architecture design.

Having a second VM is advantageous as it allows the development team to test the new control system during development without interrupting users of the existing control system. Once the first version of the new control system is completed the existing system will be phased out – initially it will be available as a fall-back system before being removed completely.

IMPLEMENTATION

To be able to operate the two systems in parallel it was necessary to find a solution for integrating the LabVIEW VIs and ICP into an EPICS system. It was decided that the most efficient means of integrating VIs in to the new system was to create an EPICS IOC (Input-Output-Controller) which communicates with the corresponding VI. To enable this, an in-house Asyn-based driver [9] was created, called lvDCOM [10], which communicates with LabVIEW via DCOM. As lvDCOM requires no modification to the VIs and can be run on a remote PC, it provides a low risk method for exposing data to the EPICS environment. The lvDCOM driver is configured by an XML configuration file which links the Asyn driver parameters to the front panel values of the VI. A helper program was created which automates the creation of this configuration file and the associated EPICS database file. This helper program makes it quick and simple to EPICS-enable an existing VI. Another Asyn-based driver was produced to create an IOC that communicates with the ICP via DCOM.

Unlike relatively static systems such as accelerators, the equipment used on an ISIS instrument beamline can change significantly on a daily basis. For this reason, the users require a method for seeing which IOCs are running and for easily starting and stopping IOCs. In the existing system, SECI is responsible for starting the correct VIs for the equipment being used based on an XML configuration file. In the EPICS community, the application ProcServ [11] is often used for hosting IOCs as it provides telnet access to the IOC and the ability to start or stop the IOC remotely. It was decided that ProcServ would be used for starting and stopping IOCs in the new system; however, ProcServ would require a bespoke layer placed above it to choose which IOCs are available based on a configuration file. For this purpose, an IOC was produced which creates an instance of ProcServ for each IOC listed in a configuration file. The status of the IOCs can then be viewed and changed directly through EPICS. It was also required that it be possible to obtain a list of PV names for the running IOCs. To enable this, each IOC was configured to output a list of its PVs to a SQLite database [12]. The PVs can then be retrieved through EPICS via a custom Channel Access Server written in Python using PCASpy [13].

It has been decided that the control system software will be deployed on the control PCs in two ways: a read-only "production tree"; and, a read/write "development tree." The production tree will contain the official release of the software suite. The development tree will be for testing new or modified software before it can be added to the production tree. The production tree will be mirrored onto the instrument PC, so it is available even if there is a network issue. The development tree will be available via a network share.

For scripting and command line control of the instruments Python was chosen, with PyEpics [14] being used as the means to communicate with EPICS. GUIs will be developed using Control System Studio [15].

In the future, the majority of VIs will be replaced with pure EPICS IOCs. If replacement IOCs already exist in the EPICS community then these can be used, otherwise it should be simple to replace many of the VIs using the EPICS StreamDevice [16]. Of course, there may be instances where replacing VIs is not an option, for example: replacing a complicated third-party VI. In these cases, lvDCOM can be used as a simple way to incorporate these VIs into the EPICS system.

CONCLUSIONS

The migration to an EPICS-based system has been a success – the new system is currently running in parallel with the existing system.

Key successes have been:

- Operating the new system on a separate virtual machine has worked well as it has enabled the development to progress without interfering with the operation of the existing system. In the future, once the existing control system is retired, the instrument control system will revert back to a single VM.
- The development of lvDCOM has made it quick and easy to produce IOCs that represent LabVIEW VIs. This has saved significant development resources as existing VIs did not need to be replaced with standard IOCs immediately, enabling development to concentrate on higher level control/design issues.
- The project has been able to take advantage of existing EPICS-based software, such as ProcServ, PCASpy, PyEpics and Control System Studio. In addition, a number of community-developed IOCs have already been identified that will be used to replace some of the VIs currently being used.
- Adopting the Scrum methodology gives the project the flexibility required to balance existing support commitments with developing the new control system.

Minor issues:

- As most institutions predominantly use non-Windows systems for EPICS it has meant that some EPICS-related packages have required modification for them to work correctly on Microsoft Windows.

ISBN 978-3-95450-139-7

- EPICS is very flexible and usually there are multiple ways of solving a problem, this often means that different institutions have developed alternative solutions. In these cases it can be time-consuming to determine the best solution.

REFERENCES

[1] http://www.isis.stfc.ac.uk
[2] http://www.aps.anl.gov/epics/
[3] http://en.wikipedia.org/wiki/OpenVMS
[4] http://msdn.microsoft.com
[5] http://www.opengenie.org/
[6] http://www.diamond.ac.uk
[7] http://neutrons.ornl.gov
[8] http://www.scrumalliance.org/
[9] http://www.aps.anl.gov/epics/modules/soft/asyn/
[10] http://epics.isis.stfc.ac.uk/
[11] http://sourceforge.net/projects/procserv/
[12] http://www.sqlite.org/
[13] https://code.google.com/p/pcaspy/
[14] http://cars.uchicago.edu/software/python/pyepics3/
[15] https://github.com/ControlSystemStudio/
[16] http://epics.web.psi.ch/software/streamdevice/

ISBN 978-3-95450-139-7