IBEX - an EPICS based control system for the ISIS pulsed neutron and muon source

F A Akeroyd¹, K V L Baker¹, M J Clarke¹, J R Holt², G D Howells¹, D P Keymer¹, T Löhert¹, C M Moreton-Smith¹, D E Oram¹, A Potter², I H Rey², T A Willemsen² and K Woods²

¹ ISIS Pulsed Neutron and Muon Source, Rutherford Appleton Laboratory, Science & Technology Facilities Council, Harwell Campus, DIDCOT, OX11 0QX, UK
² Tessella, 26 The Quadrant, Abingdon, OX14 3YS, UK

E-mail: freddie.akeroyd@stfc.ac.uk

Abstract. The ISIS pulsed neutron and muon source is currently migrating its instrument control system from a mainly locally written system, called SECI, to one based on EPICS for device control and using Eclipse RCP and Control System Studio for the user interface.

The new control system, called IBEX, was initially deployed on two new instruments, but recently the first existing instruments have been converted from SECI to IBEX. For compatibility with our existing business infrastructure, to ease migration and also to allow dual running, we have continued to use Microsoft Windows as our underlying operating system. We will present an overview of IBEX and its architecture, current status and our future development and rollout plans.

1. Introduction
The ISIS pulsed neutron and muon source [1] is a world-leading centre for research in the physical and life sciences and currently has over thirty beamline instruments. A key component in the successful delivery of experiments is the software and hardware used to control these beamline instruments.

As detailed previously [2] we are moving from a locally developed LabVIEW [3] based system, known as SECI (Sample Environment Control Interface), to one based on EPICS [4]. EPICS is used at many large facilities around the world, including the Diamond Light Source (on the same site as ISIS), the European Spallation Source (ESS) and the Spallation Neutron Source (SNS) at ORNL. As well as providing a solid software framework, adopting EPICS provides opportunities for collaboration and sharing that were not possible with the old control system.

2. General approach
As ISIS has many established beamlines, introducing a new control system must be done carefully to avoid lost beam time. Instrument scientists must also feel that the new system is both reliable and tested. To make this transition as smooth as possible the following approach has been taken:

- Continue to use Microsoft Windows as the underlying operating system [2]. Linux is more common at other facilities, but there is less expertise and familiarity at ISIS and we would run the risk of changing too much at once. We also maintain direct compatibility with existing business services (such as archiving, backup and neutron data acquisition), more easily use existing
LabVIEW drivers when necessary, and have the option of reverting back to SECI if needed. The IBEX system has, however, been developed without excluding a move to Linux at a future date.

- Develop the software using Continuous Integration (CI) managed via Jenkins, which runs both system and unit tests and creates a deployment package.
- Make use of device emulators for testing, we are using the LeWIS [5] Python framework which was developed separately as part of ISIS-ESS in-kind work.
- Make use of EPICS within the existing SECI system, both as a way to avoid duplication when new devices software is required, but also to increase user confidence. The CALab [6] package allows easy creation of a LabVIEW EPICS client. An upgrade of the muon front end magnet control was carried out this way, the ultimate requirement for shared control and monitoring across several beamlines was a natural fit for the EPICS client-server model.
- Provide LabVIEW integration tools. We have a great deal of existing LabVIEW controlled equipment, for much an EPICS driver either already exists or can easily be created using e.g. StreamDevice [7]. We do, however, have third party equipment with vendor supplied LabVIEW control, in these cases creating a clean interface by incorporating the equipment rather than rewriting control in EPICS is better. For such situations we have developed LVDCOM [8] (for LabVIEW front panel control) and NETSHRVAR [9] (for Network Shared Variable [10] control).
- Allow scientists to quickly create a simple device driver. Sometimes a new piece of equipment needs to be connected at short notice, the SDTEST package has been used by them to quickly implement an EPICS StreamDevice driver by defining simple EPICS macros in a text file.

3. Hardware architecture
This has been covered in more detail previously [2]. Though EPICS allows greater potential for distributed control with IOCs running on multiple computers, most IBEX systems currently run all IOCs on a single Virtual Machine (VM). The exception is the IMAT beamline where two neutron imaging cameras have their own dedicated PCs for camera control, these PCs run a “cut down” version of the IBEX server to allow control from the central VM instance.

4. Software architecture

4.1. IBEX server
Individual devices are controlled via EPICS IOCs running on the loopback network interface, each wrapped within a procServ [11] process harness instance. Two CS-Studio [12] archive engines record values of interest from IOCs, and are backed by a MySQL instance which is also used for storing log messages generated from IOCs [2].

Each IOC also has an associated config.xml file specifying configuration parameters (EPICS macros) it supports. These files are assembled at system build time and served to the IBEX GUI, they describe both available parameters and parameter validation rules.

On starting each IOC writes a list of its Process Variables (PVs) and associated EPICS DB info records to the instrument MySQL instance, this is then used by the GUI for PV browsing/lookup. To help with user selection, a high/medium/low “interest” level can be assigned to PVs via the DB info record and used for filtering. The info records are also used for configuring the archiving of PVs.

The neutron data acquisition is controlled by an IOC that communicates with the same program used by the SECI system. This IOC also runs an instance of the EPICS PCAS software so is able to dynamically create process variables to serve detector spectra. We are thus able to serve data from any detector pixel as an EPICS waveform record without having to pre-create a potentially large number of EPICS process variables. On instruments using event mode neutron data collection the IOC can now make use of plugins from the EPICS areaDetector framework [13] to serve integrated images to clients for visualisation during an experiment.

Running alongside the IOCs is the BlockServer process, which is the main manager process of IBEX. Its job is to manage which IOCs should be running at any particular time, and it does this by
allowing the user to create instrument configurations. The BlockServer is based on the EPICS PCASpy package [14], which allows it to serve values to EPICS clients via Channel Access.

4.2. Instrument configurations and blocks
The equipment used on a beamline can vary, so it is important to specify which IOCs should be active. Leaving IOCs running when equipment is not present can lead to large log files or unnecessary errors being flagged to users. This collection of IOCs (and any necessary parameters, such as EPICS macro and default PV values) is known as a configuration and represented as XML. An aggregation style model is supported, where a configuration can reference smaller parts called components. Thus an “instrument base” component can be created for the fixed beamline equipment and various smaller components for transient sample environment equipment. A configuration can then be assembled using relevant components for the particular experiment type being performed.

A configuration also specifies “Blocks”, which are scientist friendly names for particular EPICS values important to the experiment e.g. a SampleTemp block representing the “SOME:READBACK” IOC device PV. Blocks can have a logging frequency set and values are automatically recorded in the NeXus [15] format data file created at experiment end. A block may also be given a valid range and put under “run control”, which means that neutron data collection will automatically be suspended while it is out of range.

4.3. IBEX client
The main GUI is the “IBEX Client” (Figure 1) which is an Eclipse RCP based program. It has a similar layout to SECI, but also incorporates a new central synoptic display. A Python scripting window running the ISIS “GENIE Python” command set for instrument control is also provided.

The synoptic provides an overall view of the instrument state and key values can be displayed along with the icons. It also supports “drill down” to further screens that contain CS-Studio BOY OPIs [12] for individual devices. Multiple synoptic views can be defined, providing different views of the
instrument – these are created using a simple drag and drop editor. It is also possible to create e.g. a synoptic displaying sample environment values across several beamlines.

A recent addition to the GUI is the concept of “device screens”. Previously it was not possible to open a CS-Studio BOY OPI unless it was connected to a synoptic icon. It is now possible to define these views and associated OPI parameters separately.

As the client is configured from the server, it supports switching to view a different instrument – this has proved useful for checking the status of multiple instruments from a technician’s desktop.

5. Status and future plans

IBEX has successfully been used to commission two new instruments, and is now also running on two established instruments. We are in the process of rolling out IBEX to more instruments, and hope to complete the majority of beamlines within the next two years. As with all new software development programmes, we have experienced challenges with new hardware types and commissioning existing hardware already operational on beamlines. To mitigate these issues, we have employed device simulation as well as ensuring close communication with instrument scientists at the facility.

Currently user control scripts are run from a Python terminal on the main instrument computer. A table based script generation tool (MkScript) is available, but there is currently no direct feedback from the script itself on how far it has progressed. We are currently investigating using the nicos-daemon part of the Nicos instrument control package [16] as a script server. Work on this is progressing well, an ActiveMQ [17] proxy has been created to allow easier integration of Nicos with the IBEX control system. Nicos provides direct feedback to the client GUI as to where in a script is currently executing, as well as options to modify future operations.

As part of separate ESS in-kind work, Apache Kafka has been used for live event streaming [18]. The latest ISIS data acquisition hardware supports a network based interface and Linux drivers for this are available, so plans are being developed to update the ISIS neutron acquisition software chain which could lead to a fully Linux based IBEX instrument prototype in the near future.

References

[1] The ISIS Pulsed Neutron and Muon Source URL http://www.isis.stfc.ac.uk/
[2] Clarke M J et al 2015 ICALEPCS 2015 Conference Proceedings 0-3 URL http://accelconf.web.cern.ch/AccelConf/ICALEPCS2015/papers/mopgf048.pdf
[3] National Instruments LabVIEW Software URL http://www.ni.com/labview/
[4] EPICS: Experimental Physics and Industrial Control System URL http://www.aps.anl.gov/epics/
[5] LeWIS stateful device simulator URL http://lewis.readthedocs.io/en/latest/
[6] CALab software package URL https://www.helmholtz-berlin.de/zentrum/locations/it/software/exsteuer/calab/index_en.html
[7] EPICS StreamDevice software URL http://epics.web.psi.ch/software/streamdevice/
[8] LVDCom software URL https://github.com/ISISComputingGroup/EPICS-lvDCOM/
[9] NETSHRVAR software URL https://github.com/ISISComputingGroup/EPICS-NetShrVar/
[10] National Instruments Network Shared Variables URL http://www.ni.com/white-paper/5484/en/
[11] Procserv Process Server URL https://sourceforge.net/projects/procserv/
[12] Control System Studio URL http://controlsystemstudio.org/
[13] EPICS areaDetector URL http://cars9.uchicago.edu/software/epics/areadetector.html
[14] PCASpy: Portable Channel Access Server in Python URL https://pypi.python.org/pypi/pcaspy/
[15] Könnecke M et al 2015 J. Appl. Cryst. 48 301-305 ISSN 16005767
[16] Nicos Control System URL http://cdn.frm2.tum.de/fileadmin/stuff/services/ITServices/nicos-2.0/dirhtml/
[17] Apache ActiveMQ URL http://activemq.apache.org/
[18] Jones M D, Akeroyd F A, Arnold O, Clarke M J, Draper N, Gigg M and Moore L 2017 ICANS XXII J. Phys.: Conf. Ser. (in press)