ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-TUPV048

UPDATES AND REMOTE CHALLENGES FOR IBEX, BEAMLINE CONTROL AT ISIS PULSED NEUTRON AND MUON SOURCE

F. A. Akeroyd, K. V. L. Baker, L. Cole, J. R. Harper, D. Keymer, J. C. King, T. Lohnert, A. J. Long, C. Moreton-Smith, D. Oram, B. Rai, Science and Technology Facilities Council, Rutherford Appleton Laboratory, Chilton, UK

Abstract

IBEX is the EPICS based experiment control system now running on most of the beamlines at the ISIS Neutron and Muon Source, with plans to deploy to all remaining beamlines by the end of the upcoming long shutdown.

Over the last couple of years we have added support for reflectometry and muon instruments, developed a script generator, moved from Python 2 to Python 3, and continued to build on our suite of device emulators and tests. The reflectometry inclusions required the development of a framework to maintain the complex motion control requirements for that science technique.

Whilst it is desirable that IBEX is easily configurable, not all operations should be available to all users, so we have implemented functionality to manage such access.

The COVID-19 pandemic has meant we have also had to adapt to greater amounts of remote experiment access, for which we developed systems covering both IBEX and the old SECI control system.

This presentation will aim to provide a brief update on the recent changes to IBEX, as well as outlining the remote operation solutions employed.

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. The IBEX control system [2, 3] is currently replacing the previous control system, called SECI, and to date about two thirds of instruments have been converted. IBEX is a client-server based system composed of EPICS [4] for the server part, Eclipse/RCP/Control System Studio [5] for the client GUI (see Fig. 1), and utilising Python for scripting.

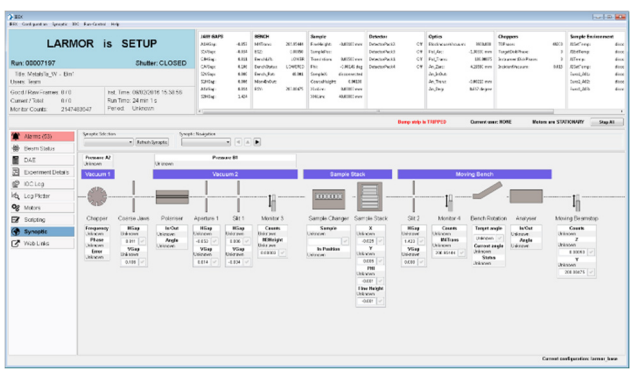

Figure 1: The IBEX user interface.

REMOTE EXPERIMENT ACCESS

Lockdowns and subsequent travel restrictions, social distancing and new safety measures brought about by the COVID-19 pandemic caused disruption to planned operation of the ISIS facility. When systems were put in place to allow the accelerator and instruments (beamlines) to operate again, it was still impossible for most external facility users to travel to ISIS to perform an experiment.

Gathering Requirements

A consultation was undertaken with instrument scientists and technicians, initially with a questionnaire and then follow up meetings with group leaders. The aim was to determine how they foresaw operations with "user not present" experiments and to look for common themes of where best to concentrate our resources. We also needed to cover both IBEX and non-IBEX (SECI) instruments with our solutions.

There were quite a range of levels of user engagement proposed, these were often determined by safety or equipment damage considerations with remote control. Only one instrument group, the muon beamlines, wished their users to have full access. Other instrument groups wished various degrees of read-only access so the user could monitor the experiment remotely, liaising with the local contact when necessary. The support technicians also wished for enhanced remote monitoring, which would allow them to better optimise timing and resourcing of general equipment tasks on site.

Full Remote Experiment Access

For the muon beamlines it was practical to allow remote users full access to the experiment equipment. We were able to achieve this by using the cloud connectivity solution provided by the commercial RealVNC package [6], effectively putting the user in the instrument cabin. The muon beamlines were not fully running IBEX at this point, so this solution also avoided having to modify the old SECI control system.

A dedicated control computer in each instrument cabin was configured for cloud VNC access, instrument scientists then added users to the relevant VNC group granting access only for the duration of their scheduled experiment. Remote users were required to use Two-Factor Authentication (2FA) [7] for access to the VNC system.

In addition the VNC system has proved useful for more than just remote user experiments. An engineer from an overseas external company was also able to take part in commissioning a new chopper system along with local staff.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-TUPV048

Though VNC was primarily used for read/write access, it can be used to grant read-only access to a computer. Such access, however, does not even let you change which window you are viewing, so is only useful if there is a particularly relevant screen that could be left visible on the system

Remote Monitoring via Web

ISIS has a "web dashboard" (see Fig. 2) for both old and new control systems, this gives a very brief snapshot of the instrument state and key values (blocks), but is more suited to quickly checking the instrument is still working or what stage an experiment has reached. To monitor progress in sufficient detail, the ability to see historical values is required.

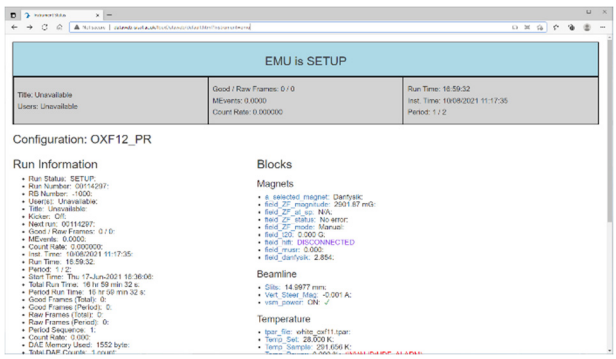

Figure 2: Instrument web dashboard.

Developing a solution applicable for both the IBEX and the SECI control system made use of a common component of both systems, the ISISICP program. This manages the fast data acquisition system, but also has access to all the sample environment data as it is responsible for writing the final NeXus HDF5 [8] experiment data file.

We chose to use Grafana [9] to provide the graphs of experiment parameters. The system is stationed behind an Apache web server proxy and uses a FreeRADIUS plugin to interact with the user office system to authenticate users and check they were part of the appropriate "virtual visit".

To provide the data for Grafana, we chose to make use of the Prometheus [10] monitoring tool as an intermediary. The common control program component was instrumented to allow it to be scraped by the Prometheus system, which then stored the values (metrics) in its time series database. A data retention period of two weeks was applied in Prometheus to the scraped data. As Prometheus is a full monitoring system it also supports defining rule-based alerts, but as we already had an existing system for this it was not something we investigated.

Figure 3: Plot of a block from the web dashboard.

A simple single value plot from following a link to a block on the web dashboard is shown in Fig. 3. A more complex interaction can be obtained via the Grafana explorer interface as shown in Fig. 4. Items monitored via Prometheus conveniently appear in the metrics dropdown list and multiple queries can be added to the same graph and then stored in the query history for use at another time.

The Grafana data view proved very popular with the various groups. The technicians also created their own cross-instrument dashboard to help with coordinating work.

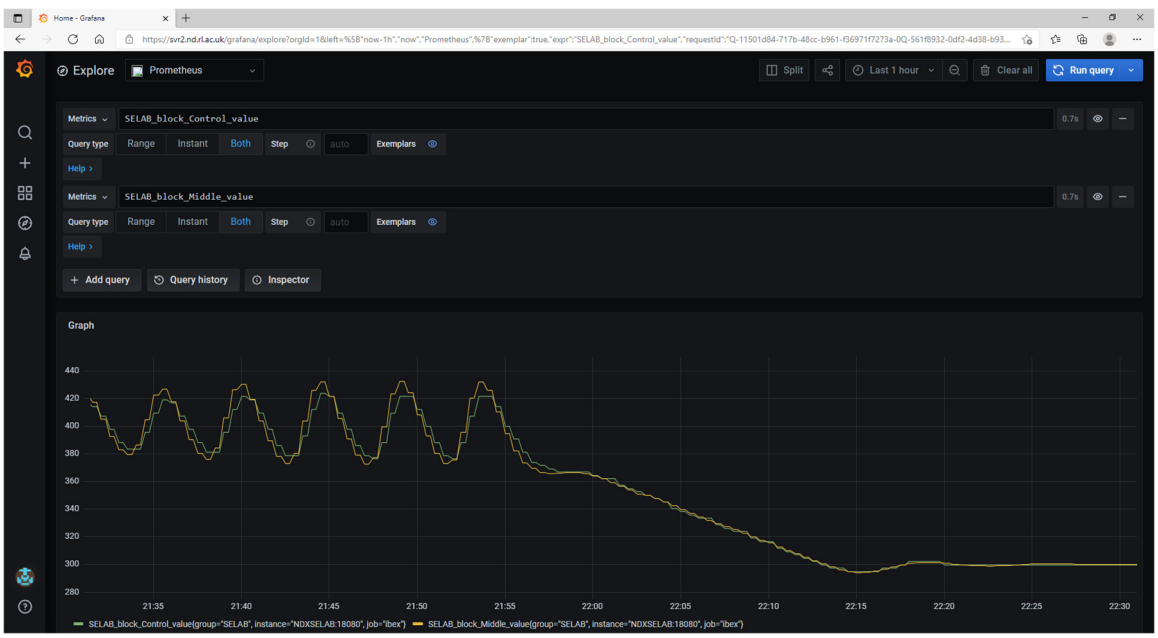

Figure 4: Grafana explorer view.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-TUPV048

Remote Monitoring via IBEX Client GUI

Many ISIS users were already making use of IDAaaS (ISIS Data Analysis as a Service) [11] for carrying out data analysis using software such as Mantid [12]. The IDAaaS system allows authenticated users to provision a new pre-configured Linux virtual machine which has appropriate resources and pre-installed software for a particular data analysis technique.

Though the IBEX control system primarily runs on Microsoft Windows, this is only (currently) a requirement for the server part of the system. The IBEX client is Java based, making use of Control System Studio and Eclipse/RCP, so is cross platform. By providing the IBEX client on the IDAaaS system we were able to give users an enhanced read-only view of the experiment, they were able to open all equipment panels from the synoptic and plot values in the data browser window, even access the live detector view served via channel access through EPICS areaDetector [13]. Users were, however, not able to change values via this interface, this access was enforced via EPICS CA gateways.

The IBEX GUI did not require extensive work to be packaged and deployed to the linux IDAaaS system. Most issues were related to accidental use of \ rather than / in pathnames and mis-matching of upper/lower case between on disk and in software filenames. These issues were corrected and automated checks added to our system.

BUILDING AND DEPLOYING SYSTEMS

The IBEX control system runs on Windows virtual machines, which provides a good business fit with the other IT infrastructure on site. We have been looking to improve the way we deploy and upgrade the system, both the IBEX applications and the Windows operating system.

Over time the performance of an operating system can suffer from repeated installation/removal/upgrade of applications and security patches; in additional different instruments can diverge in the type and version of software installed. Ideally, we would like to refresh the operating system while preserving the rest of the system.

Our new Windows 10 instrument system is generated by using the Microsoft Deployment Toolkit (MDT) [14] to build a system image with relevant applications installed, and then to attach the remaining parts of the IBEX control system as separate Virtual Hard Disks (VHDs) that have been separately created elsewhere. Applications that make use of the windows registry are installed as part of the base Windows system image, however IBEX and most of its associated utilities are purely file based and can be deployed on the non-operating system VHDs. Besides the main Windows system VHD, we have four others that are attached to the virtual machine:

- APPS: main applications minus any instrument specific settings e.g. EPICS Input/Output Controllers (IOCs)
- SETTINGS: instrument specific details and configuration information, referenced by IOCs on APPS VHD
- SCRATCH: temporary storage, raw data cache
- VAR: MySQL database files, other dynamic settings

By using MDT we have embedded the knowledge of building and configuring the control system computer into version controlled scripts. Performing a major windows and applications upgrade would now be easier and involve generating a new windows base image, mounting an updated APPS VHD, mounting the other existing VHDs, and running the upgrade script provided on the new APPS VHD which will make any required changes to files on the SETTINGS and VAR VHDs.

REFLECTOMETRY SERVER

One of the most complicated scientific techniques from the controls perspective is reflectometry. These beamlines have many axes of motion and require precise alignment as well as coordinated motion of components. Items commonly found on other beamlines, such as slits, often have an additional degree of motion on reflectometers, and super mirrors can alter the beam direction, requiring downstream components to compensate. To handle this, we developed an additional abstraction above the EPICS motor layer called the reflectometry server [15], the GUI for this is shown in Fig. 5

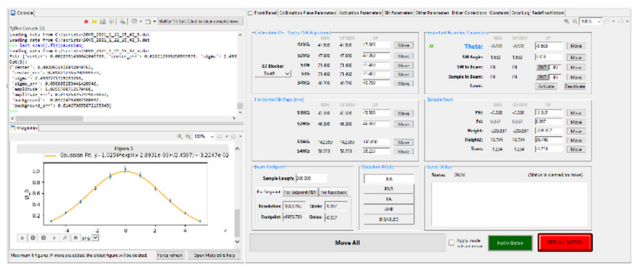

Figure 5: The reflectometry GUI.

The reflectometry server is a Python program based on the PCASpy [16] package and is configured via a Python file that defines the beamline layout. The beamline is assembled out of building blocks called components, which represent a node of interaction with the beam (either passively tracking it or actively affecting it). These components are linked to low level motors, but a management layer can provide service such as altering motor velocities to enable synchronisation of component motion and calculating a beam footprint.

The reflectometry server has recently been extended to support a bench component, which transforms three linear motion axes to motion on an arc around the sample. This has been successfully deployed to the ISIS POLREF instrument. Other improvements to the system have included:

- Allowing components multiple parking positions to choose from, these can depend on beam angle so ensuring they do not block the neutron beam
- Parking components in a sequence of multiple moves to avoid collisions
- Components can now have a variable rather than fixed Z (along beam) position, and this can be scanned.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-TUPV048

SCRIPT GENERATOR AND SERVER

Automated control of user experiments is achieved via scripts written in Python, which while powerful can be error prone. Users' scripts loaded into the system are checked with Pylint [17], which spots some errors and python 2/3 incompatibilities, but will still miss issues like referring to non–existent equipment or waiting for too much time or beam current. Other problems can occur if a user creates two scripting windows and accidentally launches two competing scripts. Running a script in a dry-run or simulation mode can spot some additional problems, but this require the script to have been written following a set of strict rules (i.e. only accessing devices via certain APIs) or else the modes will either fail or have undesired effects.

Script Generator

The IBEX script generator aims to provide a simpler table like interface for users to script the instrument. Rows are populated in a table where the columns refer to parameters that affect the control of the experiment. The parameters available and functions used to validate parameters and run scripts are configured via a local *script definition* file. More details of this system are presented in another paper at this conference [18].

Script Server

The script server is based on NICOS [19], it takes its input from the script generator and its role is to queue up scripts and provide a mechanism for interacting with the running scripts. A script server perspective is provided in the IBEX GUI; however, a future goal is to provide more dynamic interaction between the script generator and script server.

CONTROLLING USER ACCESS

The IBEX control system has several classes of user: visiting scientists, instrument scientists and technicians. Each of these groups may wish to see a slightly different view of the available information, and the ability to change specific parameters may need to be restricted by user group type.

A hierarchy of screens can be created that allows "drilling down" from a synoptic to greater details, or an "advanced" tab is sometimes used instead. Often we wish to restrict access to a screen or values on a screen, for this we have defined a system called *manager mode* that must be enabled (via entering a password in the GUI) to allow such access.

Manager mode makes use of EPICS channel access security at its heart. Process variables that require protection are placed in an EPICS access security group that only allows write access when the *manager mode* EPICS process variable is set. In addition, scripts attached to GUI panels can show/hide or enable/disable widgets based on the status of *manager mode*.

AUTOMATED TESTING

The development of IBEX utilises modern software engineering techniques, such as Continuous Integration [20], and we make extensive use of automated testing. We have developed a framework for testing our software against device emulators [21], the latter usually developed using the LeWIS package [22]. We also use Squish [23] for user interface testing.

This approach has continued to be useful to us, allowing us to spot issues and incompatibilities early when software or packages are changed.

CONCLUSIONS

The rollout of the IBEX control system to new instruments has been partly interrupted by the COVID-19 pandemic, but this also gave us the opportunity to explore and develop new remote access mechanisms that will be useful in future. The various systems and procedures that have been put in place to develop IBEX, such as extensive testing and use of emulators, taken together with the systematic creation of the instrument virtual machine system via MDT, should provide us with a good platform for future longevity and reliability.

REFERENCES

[1] The ISIS neutron and muon source, Oxfordshire, UK, https://www.isis.stfc.ac.uk/

[2] K. V. L. Baker *et al.*, "IBEX: Beamline Control at ISIS Pulsed Neutron and Muon Source", presented at the 17th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper MOCPL01, pp. 59-64. doi:10.18429/JACoW-ICALEPCS2019-MOCPL01

[3] F. A. Akeroyd *et al.*, "IBEX – an EPICS based control system for the ISIS pulsed neutron and muon source," 2018 J. Phys.: Conf. Ser. 1021 012019 doi:10.1088/1742-6596/1021/1/012019

[4] Experimental Physics and Industrial Control System (EPICS), https://epics-controls.org/

[5] Control System Studio, https://controlsystemstudio.org/

[6] RealVNC, https://www.realvnc.com/

[7] Two-Factor Authentication (TFA), https://en.wikipedia.org/wiki/Multi-factor_authentication

[8] M. Könnecke *et al.*, 2015 J. Appl. Cryst. 48 301-305 ISSN 16005767.

[9] Grafana, https:// grafana.com/

[10] Prometheus, https://prometheus.io/

[11] F. Barnsley *et al.*, "Building a prototype Data Analysis as a Service : the STFC experience.", NOBUGS 2016 Proceedings, doi: 10.17199/NOBUGS2016.65

[12] O. Arnold *et al.*, "Mantid—Data analysis and visualization package for neutron scattering and μSR experiments", Nuclear Instruments and Methods in Physics Research Section A, Volume 764, 11 November 2014, Pages 156-166, doi:10.1016/j.nima.2014.07.029

[13] EPICS areaDetector, https://github.com/areaDetector

[14] Microsoft Deployment Toolkit (MDT) https://docs.microsoft.com/en-us/windows/deployment/deploy-windows-mdt/get-started-with-the-microsoft-deployment-toolkit

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-TUPV048

[15] T. Löhnert, A. J. Long, and J. R. Holt, "Generalising the High-Level Geometry System for Reflectometry Instruments at ISIS", presented at the 17th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper WEPHA091, pp. 1300-1303. doi:10.18429/JACoW-ICALEPCS2019-WEPHA091

[16] PCASpy: Portable Channel Access Server in Python, https://pcaspy.readthedocs.io/en/latest/

[17] Pylint, https://www.pylint.org/

[18] J. C. King *et al*., "The IBEX Script Generator", presented at ICALEPCS'21, Shanghai, China, Oct. 2021, paper TUPV049, this conference.

[19] NICOS, https://nicos-controls.org/

[20] Continuous Integration, https://en.wikipedia.org/wiki/Continuous_integration

[21] T. Löhnert *et al*., "Testing Tools for the IBEX Control System", presented at the 17th Int. Conf. on Accelerator and Large Experi-mental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper WEPHA090, pp. 1295-1299. doi:10.18429/JACoW-ICALEPCS2019-WEPHA090

[22] LeWIS – Let's Write Intricate Simulators, https://github.com/ess-dmsc/lewis

[23] Squish, https://www.froglogic.com/squish/