High Power Laser Science and Engineering, (2023), Vol. 11, e56, 25 pages.
doi:10.1017/hpl.2023.49

HIGH POWER LASER SCIENCE AND ENGINEERING

REVIEW

Control systems and data management for high-power laser facilities

Scott Feister[1], Kevin Cassou[2], Stephen Dann[3], Andreas Döpp[4], Philippe Gauron[2], Anthony J. Gonsalves[5], Archis Joglekar[6,7], Victoria Marshall[3], Olivier Neveu[2], Hans-Peter Schlenvoigt[8], Matthew J. V. Streeter[9], and Charlotte A. J. Palmer[9]

[1] *California State University Channel Islands, Camarillo, California, USA*

[2] *Université Paris-Saclay, CNRS/IN2P3, IJCLab, Orsay, France*

[3] *Central Laser Facility, STFC Rutherford Appleton Laboratory, Didcot, UK*

[4] *Ludwig–Maximilians–Universität München, Garching, Germany*

[5] *BELLA Center, Lawrence Berkeley National Laboratory, Berkeley, California, USA*

[6] *Department of Nuclear Engineering and Radiological Sciences, University of Michigan, Ann Arbor, Michigan, USA*

[7] *Ergodic LLC, San Francisco, California, USA*

[8] *Helmholtz-Zentrum Dresden – Rossendorf, Dresden, Germany*

[9] *Queen's University Belfast, Belfast, UK*

(Received 23 February 2023; revised 13 May 2023; accepted 30 May 2023)

Abstract

The next generation of high-power lasers enables repetition of experiments at orders of magnitude higher frequency than what was possible using the prior generation. Facilities requiring human intervention between laser repetitions need to adapt in order to keep pace with the new laser technology. A distributed networked control system can enable laboratory-wide automation and feedback control loops. These higher-repetition-rate experiments will create enormous quantities of data. A consistent approach to managing data can increase data accessibility, reduce repetitive data-software development and mitigate poorly organized metadata. An opportunity arises to share knowledge of improvements to control and data infrastructure currently being undertaken. We compare platforms and approaches to state-of-the-art control systems and data management at high-power laser facilities, and we illustrate these topics with case studies from our community.

Keywords: big data; community organization; control systems; data management; feedback loops; high-power lasers; high repetition rate; metadata; stabilization; standards

1. Introduction

1.1. Shifts in high-power laser technology necessitate revised digital infrastructure

High-power and high-intensity laser–plasma interactions provide a versatile experimental platform. They can produce extreme plasma environments, either for laboratory astrophysics and fundamental plasma physics, or as a unique source of secondary radiation. Secondary sources include bright, keV–MeV X-rays[1], low-emittance and high-current electron beams[2–4], GeV electron beams[5–8], ultra-short MeV proton beams[9,10] and pulsed neutron sources[11]. These sources have demonstrated significant potential for applications[12] including rapid, high spatial resolution X-ray tomography[13–15], free-electron lasing[16–18], FLASH radiotherapy[19,20] and materials damage testing[21]. In order to develop these sources for applications (e.g., optimizing their stability and tunability), and in order for them to be competitive with alternative sources, it is necessary for the source repetition rate to drastically increase from sub-Hz to hundreds of Hz (and beyond).

Tackling the obstacles to achieving multi-Hz repetition-rate high-intensity laser–plasma interactions has been a focus of the high-power laser community in recent years. Great progress has been shown in laser technology[22], replenishing targets[23–29] and online diagnostics[30–33]. The increasing availability of experimental facilities compatible with high repetition rates now highlights the need to adjust traditional experimental practices, and control, in order to fully exploit the opportunities offered by these systems[34]. Among these

Correspondence to: Charlotte A. J. Palmer, Queen's University Belfast, Belfast BT7 1NN, UK. Email: c.palmer@qub.ac.uk; Scott Feister, California State University Channel Islands, Camarillo, California 93012, USA. Email: scott.feister@csuci.edu

needed adjustments, new systems should enable automation of scans of experimental and laser parameters, rather than relying on repeated re-configuration through manual user input. In addition, data from a large suite of diagnostics should be acquired and collected at least at the repetition rate of the slowest experimental component (e.g., laser, target), ideally with the associated metadata to enable efficient online analysis. Such systems can then be used for automated communication between diagnostic elements and control elements, enabling control without a human 'in-the-loop'[35]. In recent years this has been demonstrated in proof-of-principle experiments[36–39], illustrating the deeper insight and source enhancement that is enabled by such a shift in methodology.

With a new generation of high-repetition-rate, high-power lasers nearing completion, re-tooling is underway in many areas linked to high-power laser facilities and their experiments. These areas include targetry, scientific instruments, data pipelines, mechatronics systems, analysis software, data pipelines and data management. Recently, two special issues of peer-reviewed journals have been dedicated to 'target fabrication'[40] and 'the high repetition rate frontier in high-energy-density physics'[41]. This manuscript addresses a related topic – facility control systems and data management – which is the digital infrastructure upon which scientific experiments are designed and executed.

1.2. Approaches to tooling

An effective laboratory control system can enable facility-wide communication among instruments, sensors, software and humans. To clarify each category in the context of high-power laser facilities, (1) instruments at high-power laser facilities could include, for example, laser power meters, laser contrast diagnostics, particle time-of-flight detectors and electron spectrometers; (2) sensors could include, for example, thermometers and pressure sensors; (3) software could serve the roles of, for example, laboratory automation, data storage, human interfacing and online data analysis; (4) the humans involved could include, for example, facility staff, principal investigators, graduate students, postdoctoral scholars and visiting scientists.

One common approach to control in a high-power laser facility is to leave various elements semi-isolated from one another, with humans as the direct mediators of communication between various facility elements. Elements may be computer-interfaced, for example, through vendor-provided graphical software or through a custom LabVIEW interface, but not in digital communication with one another. In this approach, a person is responsible for synthesizing data inputs from various elements, and making the appropriate system re-configurations between repetitions (i.e., laser shots) (see Figure 1(a) for an illustration). The increase in the repetition rate of high-power laser facilities necessitates a need to move beyond this human interfacing, towards digital interfacing. This leads to the adoption of control methods that are managed primarily in network-interfaced software (which itself is configured by humans) (see Figure 1(b)). A distributed networked control system[42] is one in which the laboratory elements interact between one another (rather than to one central person or even one central control hub), and it is designed to be fault-tolerant and scalable. It may implement strategies (such as redundancy) to ensure communication and control can continue even when confronted by network disruptions or other issues.

Data management is another area of digital infrastructure in a laboratory where the approach may be organized, disjointed or somewhere in between. One common approach to data management in a high-power laser facility is to maximize flexibility by utilizing humans at all stages of data management. For example, team members might initiate data collection on many computers separately and keep track of the relationships between the data in a laboratory notebook. Metadata – or the data that provides context for the data itself – could include information such as environmental parameters, laboratory configurations during data collection

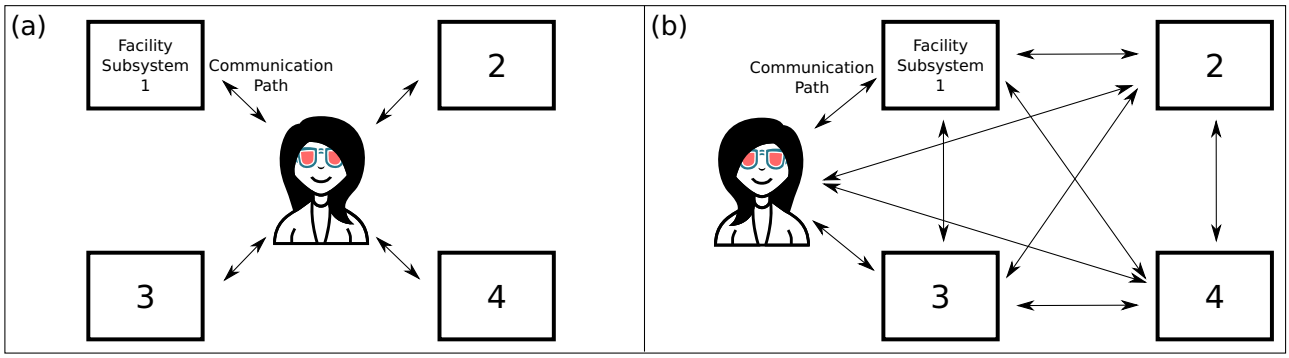

Figure 1. (a) Diagram of a high-power laser facility without a laboratory-wide control system. People tie the subsystems together. For example, a person might adjust parameters on the laser amplifier subsystem in response to observations from the target-chamber subsystem. (b) Diagram of a facility with a laboratory-wide control system. Consistent implementation throughout the facility opens new communication pathways between subsystems. These new pathways enable laboratory-wide automation and control feedback loops without human mediation. Humans retain communication with all subsystems.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

and the experimental diagnostics settings. This metadata is manually captured and recorded by a human operator, then collected together with the data and labelled in such a way that it can be re-associated with the relevant data. In this approach, when other scientists wish to leverage that data in analysis, people communicate the data and metadata to collaborators via email, shared cloud folders or other means.

This approach to data management for high-power laser facilities has relied on sufficient and accurate human-described contexts associated with each element of data. However, when lasers are pulsed/fired at many times per hour, less time is available to capture the metadata – and thus, there is more opportunity for mislabelled or missing data. Human error is also an issue in low-repetition-rate experiments, and so automated recording of metadata can benefit even these less-frequent measurements. However, in the most extreme cases of new high-power laser systems capable of generating terabytes to petabytes of data in a span of days or weeks, it is not even feasible to copy the raw data onto a personal computer, let alone easily distribute it in-full to colleagues. Isolated, fragmented copies of partial datasets could be expected to result from person-to-person distribution. We suggest that organized approaches to data management for the next generation of high-power laser facilities involve people stepping back from direct manipulation and distribution of data. So, we propose approaches to data management with systems that (1) reliably save data with unambiguously associated metadata; (2) provide easy access to the data from multiple users and institutions; and (3) include modular software tools for analysing data.

Many laboratories in our community have already implemented facility-level approaches to control systems and data management. We share several examples as case studies throughout this paper, and hope that the community can seize this opportunity to learn collaboratively and maximize the utility of the next generation of digital infrastructure.

1.3. Case study: the Draco laser at Helmholtz-Zentrum Dresden–Rossendorf

This community example adds concreteness to the challenges of upgrading existing digital infrastructure, which may be disparately organized.

How did multiple subsystems develop at this facility? Helmholtz-Zentrum Dresden–Rossendorf (HZDR) was originally founded as a nuclear physics research centre in mid-1950s and hosts nowadays, together with several other facilities, the Draco laser system[43], in operation since 2007, and the PEnELOPE system[44,45], in commissioning. Research into relativistic plasmas was initiated at HZDR in the mid-2000s. Laser systems were installed for synergy reasons as additions to the existing accelerator-driven photon and particle source called ELBE[46], sharing structural shielding and building infrastructure, as well as the radiation monitoring and interlock system of ELBE. Further control systems are complete additions due to the higher flexibility of a laser–plasma experimental arrangement compared to a fixed accelerator machine. These systems were either newly developed or were part of the corresponding subsystem itself (e.g., the laser control system by Amplitude and Scarell). The newly developed components were conceived as stand-alone systems for the swift start of operations. Due to the external development of the laser control system, an approach in a single framework was not possible or desired. Instead, subsystems were developed step-by-step in a historical progression, partly in parallel and partly by the vendor.

Subsystems were each developed to suit different needs. For example, the vacuum control system in a laser–plasma experiment can have less strict interlock requirements than for a superconducting accelerator like ELBE, but it must allow for frequent pump-down cycles with a simple user interface. Likewise, in contrast to the more long-term fixed setups found at a typical accelerator facility, actuators in laser–plasma experiments are frequently re-configured and assembled with new components.

Despite systems being based on different frameworks, subsystems are partly interfaced to one another, and re-usability and uniformity within a subsystem, where possible, is implemented. As an example thereof, the laser–plasma data acquisition (DAQ) is often done with cameras. The software Laser Light Inspector[47] provides a common user interface for various camera vendors with a number of live analysis features, and was therefore chosen by HZDR for camera acquisition control. It further provides a centralized interface for remote control of the camera clients. This reduces efforts for, for example, file path configuration, significantly since the vast majority of detectors are operated via this software, which leads to a high level of homogeneity within this subsystem.

Likewise, acquired images are always stored locally at each client, like the native vendor software does for other detectors, such as optical spectrometers or oscilloscopes, not being integrated in Laser Light Inspector. As a result, dozens of acquisition PCs are deployed, each operating a few cameras and/or other detectors and collecting data locally for all detector types. Although imperfect, this methodology does offer high flexibility and redundancy, since virtually any camera or device can be connected to any nearby PC. Locally stored data files are regularly copied over the network to a central data repository by file synchronization software[48].

How do humans execute and process each shot? The above operation mode, involving dozens of dedicated acquisition computers, has led to quite efficient subsystems with a high degree of flexibility, but requires significant human interaction and consequently imposes a high work load on scientists. There is no assigned staff operator crew to tie together the local subsystems as one would find at a larger accelerator facility.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

Figure 2(a) illustrates the flow of operations for laser-driven ion acceleration experiments, typically done at the shot-on-demand level, that is, about 1 shot per minute, and the parameters are deduced from previous shots. Laser-driven electron acceleration experiments operate similarly but typically perform a series of shots at one point in parameter space with repetition rates between 0.1 and 0.5 Hz, before changing the parameters.

In the shot preparation phase, the numerous subsystems are prepared by staff and scientists, partly in parallel. This includes evacuation of experimental chambers, initializing the detectors, searching and (inter)locking of the experimental rooms and preparation of the laser to operate at high pulse energy.

The actual high-power laser shots are performed as a sequence of steps (shown as a large circle in the centre of Figure 2(a) and enumerated 1–7), coordinated by the scientists, which partly transfer status information between the subsystems but also process acquired information and plan the course of action. First, the diagnostics need to be checked and armed (e.g., save on trigger, correct destination file path) and (2) the experiment must be in the correct positional state (e.g., correct target and detector settings). Then all facility and safety interlocks (3 and 4) must be fulfilled. If everything indicates a 'go', a click in the laser control software starts a shot sequence (5).

The laser shot sequence consists essentially of a well-timed series of trigger signals (6), controlling processes within the laser system (e.g., flash lamps and Pockels cells) to produce the energetic laser pulse as well as triggering all detectors at the experiment with the appropriate delay such that they can record the ultrafast interaction. That trigger signal from the laser to the experiment (blue dotted line in Figure 2(a)) is so far the only direct and automatic information transfer between subsystems that is not mediated by scientists.

After that sequence, the immediate results can be viewed (7) and the cycle can start again with a new target and refined detector parameters. This step is not so much information transfer between the systems like the other steps, but rather information processing and decision making, where scientists are mandatory.

As shown in Figure 2(a), subsystems do have automatic saving or logging, but since they are all autonomous, each subsystem's storage is independent of the others and they therefore need some relation to each other. In addition, not everything can be automatically logged, for example, observations and reasonings. Hence, there is the need for a second layer of schematics for manual logging and tying relationships. This is depicted in Figure 2(b), which complements the right-hand part of Figure 2(a), where the primary experimental data storing is shown; data from the experimental environment, such as the vacuum system, is rather secondary in this regard. Figure 2(b) shows, in particular, the importance of manual logging, that is, documenting all work of the setup and calibration in an electronic lab notebook (ELN), as well as documenting the shots in a shot log. The shot log is very important for later analysis of experiments as it documents the key experimental parameters and observations, for example, diagnostic filter settings or unexpected laser behaviour. It also provides, via shot counts and timestamps, the relations between the automated logs. Finally, the connection to further metadata, including descriptors for the entire experimental run, happens at that manual stage. However, those relations are to a large extent only visible for humans.

The ELN, realized by a Mediawiki system[49], allows for the full and rich documentation of the facility, in particular the setup, development, maintenance and calibration works, but can also store or link the experimental shot data.

Are there plans to move towards a facility-level control system? HZDR's laser experiments deploy a number of different control systems, based on various software frameworks, partly for historic reasons. The ansatz of HZDR is to arrange the subsystems in a coherent way, rather than re-building the system in a single framework. Coupling is currently established via scientists and interfaces, and will always be for the sake of flexibility. The goal for developments is rather to add interfaces to the subsystems where necessary and to add agents in between, as shown in Figure 1, such that more and more automation can take place where the established workflows allow for it. HZDR respects the fact that there will always be some new development that should be quickly embedded into experiments; hence, the systems should have interfaces at very general, low-level tiers, such that scientists can realize the changes of the control system. The following are a few examples of existing plans by this facility to reduce reliance on human interfacing between subsystems and increase coherence.

- To reduce the work load of status monitoring during the shot cycle, a handler could generate a joint signal from the vacuum control system and radiation safety system, which could be fed to the laser system to inhibit the shot sequence if either subsystem is not in the proper state.
- The laser system could generate a common shot ID that can be fed to the diagnostics. It might be necessary to add an agent that could add that ID to all file names of files generated from diagnostics. This would keep the primary acquisition routine as is and add valuable metadata to all kinds of detection.
- Upon synchronizing the files to a central repository, they could be parsed to extract metadata, which is currently encoded in the file path and name. That metadata could be sent to a database such as SciCat[50], and that database could be joined with that of the shot log. Again, such an approach would be independent of the acquisition software and therefore very general.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

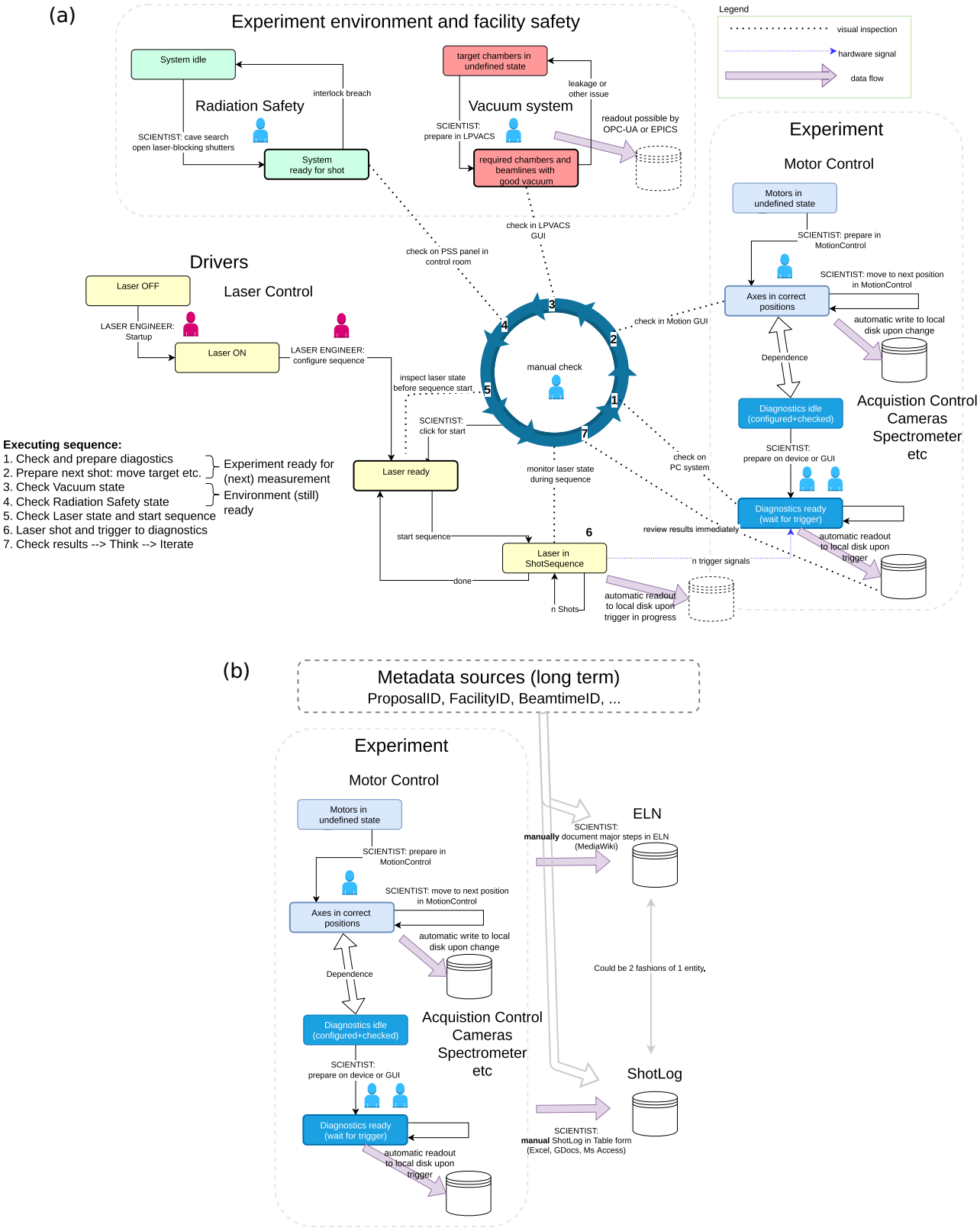

Figure 2. Engineers and scientists (represented with maroon and blue icons, respectively) manually conduct steps in the operational sequences for Draco at HZDR. (a) Facility laser shot sequence diagram for ion acceleration experiments. Note the many steps in which people tie together subsystems. The central blue circle indicates the laser shot sequence, executed and surveilled by scientists. Automated storage is part of many subsystems. (b) Manual logging to complement and complete the data storage, to curate and enrich with the metadata.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

- On individual diagnostic computers specialized analysis scripts can be developed and deployed online during the experiments. Analysed data can be forwarded via messaging protocols to a flexible database such as MongoDB[51] and visualized via Grafana[52] during experiments. This can help to literally better see changes and relations of otherwise tabular information.

These adaptations are a step in the right direction towards reducing human interfacing. They will help streamline open-loop operations at moderate repetition rates. However, without a move to a distributed networked control system, such a system is not ready for facility-level closed feedback loops. In addition, although metadata has been considered, this system relies on files passed across a network, and is not configured with high-speed data analysis pipelines for high data rates.

1.4. Digital infrastructure for next-generation facilities

The case study from HZDR shows that historically grown control systems may operate efficiently and with high flexibility at moderate repetition-rates. However, they also become increasingly complex over time, requiring significant human resources for interfacing between individual parts. Higher data rates are a necessary step forward to provide deeper insight into high-power laser–plasma interactions and to produce competitive secondary radiation sources for applications. However, moving towards increasing number of diagnostics and higher repetition rates pushes the boundaries of (too) heterogeneous approaches. Instead of being monolithic, many modern control systems are based on modular approaches, allowing them to scale up while keeping complexity at a reasonable level and allowing them to quickly adapt to changing requirements. The modularity also allows for different levels of communication, from low-level device access to high-level user interfaces. However, the choice of an adequate, future-proof control system is also a lasting one. Transitioning to a new control system may take years and requires considerable effort. Many facilities in our community are faced right now with choices that will impact their operation for the next decades, as they develop new digital infrastructure to meet the needs of next-generation high-power laser experiments. In the next sections, we compare platforms and approaches to next-generation infrastructure for control systems and data management at high-power laser facilities, and illustrate these topics with case studies from our community.

2. Approaches to laboratory control

While there are clearly available strategies for developing control systems to make full use of multi-Hz DAQ rates, realizing them can be a complex task in itself. Here (Section 2.1), we illustrate this by considering opportunities enabled by organized approaches to laboratory-wide control and several very real challenges inhibiting adoption of these new approaches within the high-power laser community (Section 2.2).

2.1. Opportunities

2.1.1. Laboratory-wide automation

When controls are operating in isolation, a human being is required to make adjustments to various systems within the laboratory. When controls are integrated into a digital control system that is organized throughout the laboratory, automation across subsystems is possible. For example, a laser intensity scan can be executed and may involve adjusting pump laser diode timings in the amplifier, translating a motorized stage in the compressor chamber, tilting a mirror in the target chamber and increasing the image gain setting on a scientific camera. Laboratory-wide automation is important for scientists because it enables execution of a planned shot sequence in which parameters are adjusted across multiple laboratory subsystems, without pausing work to wait for humans to make manual adjustments. As a consequence, automation of a shot sequence can dramatically speed up the overall scientific experiment and free up time for tasks requiring the specialist skills of a scientist.

2.1.2. Laboratory-wide feedback control loops

A major opportunity of extensive automation is to ‘close the loop’ across an entire experiment, enabling automated control based on scientific data. Data acquired from measurement of laser–plasma interactions can be piped into the control system to make control decisions. Importantly, with a control system implemented in an organized way, scientists have wide flexibility to decide which controls will be manipulated, and a common interface to make changes to the control settings. With humans out of the control loop, feedback-driven decisions can be made at much higher repetition rates and with reduced human bias.

2.2. Challenges

2.2.1. Modularity and network bandwidth management

With abstractions in place between device hardware and control interfaces, one can more easily swap out low-level hardware without breaking the high-level control interfaces. For example, one might upgrade camera hardware and switch to a different device driver, without impacting the software for display interfaces or image analysis. This leads to increased modularity of devices in the laboratory, since software for display, storage and analysis can be isolated from the complexities of device hardware.

When software systems for display, storage and analysis are separated from one another and communicating via com-

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

mon protocols, this allows for modularity in the software. For example, multiple different analysis pathways can be built that expose the same public interface, such that they are interchangeable in the experimental data pipeline, to allow for comparison and confidence in each analysis. This provides flexibility, allows for re-use of software components (e.g., data storage modules can be re-used with small adaptations in similar subsystems) and can in some ways bring more order to the system as there are fewer hidden, unknown connections between software components than that one might find in a monolithic codebase. However, it also raises the challenges of organizing various versions of these modules, and keeping track of changes to public interfaces and how this affects dependent modules, since the software modules are not kept globally in lockstep. Fortunately, this challenge is common in non-scientific software development, and there are software development best practices for managing dependencies and establishing versioning to help manage the added complexity of interacting modules. Separation of data display, storage and analysis across a local network can result in high network data rates, especially if large images are being streamed between multiple points in a local network. One approach to mitigate this issue without losing the benefits of modularity is to build modular software components, communicating via public interfaces, but have them running on the same computer (or a small local subnetwork) so that laboratory-wide network bandwidth is not an issue. A second approach is to send reduced data across the network, for example, the polynomial fitting coefficients to a curve. A third approach is to be selective about which data is rejected and which data is transferred or stored. Approaches to down-selecting data within the data pipeline, including coordination of flagging certain data as noteworthy across the entire laboratory, are utilized at many large, data-heavy facilities. For example, the LHC ATLAS experiment involves 'trigger chains' to reduce data streams through multiple levels of decimation, starting with hardware selection and ending with customizable software selection algorithms[53–55]. As a second example, the LCLS-II Data System[56] takes different approaches to configurable in-line data reduction. Although the technology developed for these experimental facilities may be oversized for many in the high-power laser community, the ATLAS and LCLS-II examples can provide inspiration for smaller-scale event-selection techniques for data rate reduction, and provide evidence that managing network bandwidth is possible even with massive quantities of high-repetition-rate data.

At the level of challenges currently faced within the high-power laser community, cameras acquiring digital images at a high repetition rate present a unique challenge to a networked control system, and require special care during control system design to avoid overloading the network with image-data traffic. For example, a single monochrome, 8-bit, 10-megapixel image may represent 10 megabits (~1.3 megabytes) of data, and streaming images from two of these cameras with no image compression at a repetition rate of 50 frames per second would consume the entire theoretical bandwidth of a 1 Gb/s network. One approach to managing cameras (without sacrificing image storage through event selection) is to store images locally, and only send limited metadata about the stored image across the network (rather than sending the full images). An elegant implementation is found in the FACET-II system of SLAC[57], in which camera data is acquired by a control computer and then saved to network-attached storage that is proximate to the control computer (the transfer occurs across only a single network switch). In this way, all heavy image traffic stays localized to the most-local network switch, and the overall network (which includes many distributed network switches) is not burdened. Similar to this approach, many single-board computers (such as the Raspberry Pi 4, NVIDIA Jetson and Beaglebone AI-64) are now powerful enough to provide sufficient data processing for a single camera image stream, with on-board GPU analysis of images, and to host large storage devices, such as solid-state hard drives or external USB hard drives. These single-board computers can be configured to act as control computer, analysis computer and network-attached storage, such that image data need not stream across the network at all, not even to a local network switch.

If common abstractions are adopted across multiple laboratories in the high-power laser community, this enables yet another opportunity to share devices across laboratories with minimal modifications to the digital infrastructure. This could facilitate travel to facilities with scientific instruments that plug into the digital infrastructure at that facility and work as they did at other facilities.

2.2.2. Learning curves and inflexibility

Unfamiliarity can impede the adoption of newer control systems, as stakeholders continue to 'use what they know'. Systems with robust software engineering foundations can require more advanced technical knowledge to develop, modify and manage. Steep learning curves can reduce flexibility, as stakeholders feel disempowered to modify their own control system. New control systems can be especially frustrating to team members who know how to build individual controls with their own tools of preference, but do not know how to integrate these into the control system. If these challenges are unaddressed, a high-quality control system architecture can inadvertently ossify the facility, serving as a barrier to entry for new devices and new team contributions.

An idea for mitigating these challenges involving the 'human factor' is to follow human-centred design[58]. In human-centred design, stakeholders are incorporated into the design process. Here, we provide one usable example of how human-centred design can be implemented to improve

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

control systems in a high-power laser laboratory. At a facility, a single person who is knowledgeable about the control system software (whether that be a dedicated software engineer, or a software-savvy graduate student) could block off one day per month in their schedule for observations. That software person could spend this one day per month observing graduate students, scientists and technicians as they perform regular laboratory tasks. To ensure operations are not disrupted, the software person would take on an important responsibility to stay 'out of the way' of stakeholders, such that the other members of the facility could continue using the facility as usual. Through observation, the software person could identify core tasks that various stakeholders need to do in the lab, which involve human–computer interaction, and keep that list up-to-date with each subsequent monthly observation. This document could be shared and asynchronously edited by stakeholders, or entirely ignored by stakeholders, depending on their preferences and schedules. The software person could furthermore document findings and adjust future versions of the software to fold in these observations. This repeated cycle of observation and improvement could result in overall improvement to the control system over the course of many months or years.

An example of a practical avenue for asynchronous communication of stakeholder needs would be building a single-click issue reporting mechanism in the control software (with this reporting system itself observed and streamlined to reduce frustration), such that issues can be created and then followed up on by the software person. Furthermore, the software person can regularly have short, informal conversations about experiences using the control system with stakeholders, and personally take on the burden of documenting these issues and experiences (rather than putting that burden on the stakeholders). Although implementing these ideas will require some time in the software person's schedule, this time will be paid back by many small improvements to daily operations for the entire team at the facility. If human-centred design is implemented at multiple facilities in our community, sharing notes at conferences can help identify issues they may not have noticed, and sharing the methods tried to solve stakeholder problems can help the community converge towards common solutions to common problems.

Science at some laboratories requires more flexibility in the control system than others, and building the same degree of system inertia into control systems at two laboratories with different science cultures will lead to different results. Trade-offs between control system rigor and ease of prototyping can be considered for different scientific facilities, bringing the conversation to the forefront with stakeholders. Human-centred design at all stages of development will help next-generation control systems meet the disparate needs of stakeholders in high-power lasers, including when compromises are required between stakeholder needs. At the facility level, as large systems are being designed, involving as many stakeholders as possible in the design stages of a control system allows for specifying better engineering requirements, which makes the control system better suited to all stakeholders.

2.2.3. *Timing and synchronization*

Avoiding data synchronization errors at high repetition rates (including 'off-by-one' data misalignment errors) requires several elements to work together flawlessly. First, each data element's timestamp must be locally precise to a significantly greater tolerance than the time between consecutive laser shots. Second, each data element's timestamp must be globally accurate (across the entire laboratory), to similar minimum tolerances. Third, any internal delays for scientific instruments must be accounted for – as an example, consider a camera stream that feeds image buffers that are off-by-one, or instruments that return data at a longer delay than others.

Fortunately, timing and synchronization problems have been deeply considered and solved at many accelerator science facilities[59–62]. The high-power laser community can learn from and adopt prior art in this area.

Two network timing synchronization protocols utilizing a facility's Ethernet infrastructure are the NTP (network time protocol), which aims to provide up to a few-millisecond precision over a wide-area network, and the PTP (precision time protocol, IE1588)[63,64], which aims for sub-microsecond precision within a single local-area network. For repetition rates up to about 1 Hz, time-stamping data using the control computer's time (synchronized via the NTP for sub-second accuracy) is sufficient. For rates up to approximately 100 Hz, and for networks with consistent ping times less than 1 ms, time-stamping can still be feasible at the control computer rather than within the DAQ hardware, provided there is consistently fast processing of a consistent data stream from the DAQ hardware. Specialized concurrent networking approaches and tools such as zeroMQ[65] can aid in synchronization.

For higher repetition rates, for example, high-power lasers operating at 1 or 10 kHz, dedicated timing hardware is commercially available[66,67], and the data is typically time-stamped at the DAQ level (rather than relying on the control-computer operating system). An alternative solution to laboratory-wide time-stamping is to leverage the trigger counting available on many DAQ devices (or to write one's own trigger counting firmware for inexpensive microcontrollers), and to maintain a global laboratory trigger count rather than a global laboratory time.

2.2.4. *Legacy instrumentation*

Existing equipment in laser–plasma facilities can present friction to the adoption of a laboratory-wide control system. This is particularly true of important legacy devices due to incompatibility with modern software or operating systems. However, this need not be the case.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

Possibilities include finding existing device drivers or using vendor-supplied software development kit, which can be used to integrate the device properly into the control system. Alternatively, the legacy user interface (possibly a vendor-provided interface) can continue to be used with a patch between the legacy software and the laboratory-wide control system. For example, a commercial user interface for a camera may already permit saving images to a file, and one could write an interface layer that reads the latest image from the file and makes it accessible on a laboratory-wide control system. A last option would be to maintain legacy devices as they are, existing apart from the control system. The latter two approaches might be especially apt for devices requiring an old, insecure Windows computer without a network connection.

3. Comparison of platforms for laboratory control

While some facilities can offer significant research software engineering to support the development of control systems, many labs, in particular university-based systems, must adapt or develop their own systems. Importantly, these must be manageable for researchers without a background in software engineering.

To date, a number of experiment control systems based either on expansion of previously existing experiment control systems at older laser facilities, or the adoption of control systems from beyond the laser community (e.g., high-energy physics and photon science) have been explored. This section reviews the advantages and disadvantages of several of these, as well as the unique challenges facing the high-power laser–plasma community in control for future high-power laser facilities through specific examples.

3.1. LabVIEW

3.1.1. Overview of LabVIEW

National Instruments (NI) LabVIEW is a graphical programming environment, and is not itself a control system. However, it is already utilized in some aspect for controls at most high-power laser facilities in our community. LabVIEW is highly accessible to beginners and experts alike, and is widely used across many science domains. One major feature of LabVIEW, in the context of control system design, is the ease with which users can create interactive graphical interfaces. A second major feature is a robust library of instrument drivers. Many instruments used at high-power laser facilities already ship with vendor-provided LabVIEW drivers. A third major feature is a built-in infrastructure for shared variables across a network. As a drawback relative to open platforms, LabVIEW has licensing requirements that must be navigated for each deployed instance. Also, since it is a graphical programming language, code written in LabVIEW does not easily mesh with schemes for text-based version control.

Given the many LabVIEW users in our community, the reader is likely familiar with many features and downsides that have not been discussed in the preceding brief overview. A very large number of existing control systems in our community are based on LabVIEW. Some are built in a consistent fashion throughout the laboratory so as to unlock all the high-data throughput, interconnected benefits of high-power laser facilities. The case study that follows illustrates how LabVIEW was incorporated into a home-grown control system architecture called GEECS at the BELLA Center.

3.1.2. Case study: BELLA Center at LBNL

Many labs, in particular university-based systems, adapt or develop their own systems. The GEECS (generalized equipment and experiment control system) provides an example of a home-grown control system developed over a number of years at the BELLA Center, which is open and modular. Here, the GEECS team describe their system and considerations for others choosing to implement a home-grown control system. At the forefront, GEECS is designed to be manageable for researchers without a background in software engineering.

What is the BELLA Center? The Berkeley Lab Laser Accelerator (BELLA) Center focuses on the development and application of laser–plasma accelerators (LPAs). It houses four Ti:sapphire laser facilities. The highest peak power laser system is the BELLA PW[68], providing up to 40 J on target in less than 40 fs with a repetition rate of 1 Hz. It has been utilized primarily for research on the high-energy physics application of LPAs[6], and more recently for ion acceleration studies and applications[69]. There are two 60 TW systems operating at up to 5 Hz, one of which couples a sophisticated electron beamline including an undulator to its LPA[70], and another with two beamlines designed for MeV photon production via Thomson scattering[71]. Finally, the BELLA kHz LPA facility consists of a few-mJ few-cycle laser system operating at the kHz repetition rate, which can produce few-MeV electrons that are of interest in a variety of applications, including medicine and security. Each of these facilities is controlled and monitored by the GEECS developed in the BELLA Center, with typically dozens of computers, hundreds of devices and thousands of process variables (PVs).

What is GEECS and why is it well-suited for the BELLA Center? GEECS is a complete software solution for control, monitoring, alarming and data logging of devices for process control and experimentation. For LPAs this means controlling laser, vacuum, timing, target and diagnostic subsystems, and synchronizing data collection for each laser pulse. The GEECS architecture is shown in Figure 3 and generally follows ANSI/IEEE-1471-2000[72], as described in Ref. [73]. The GEECS framework is scalable, distributed

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

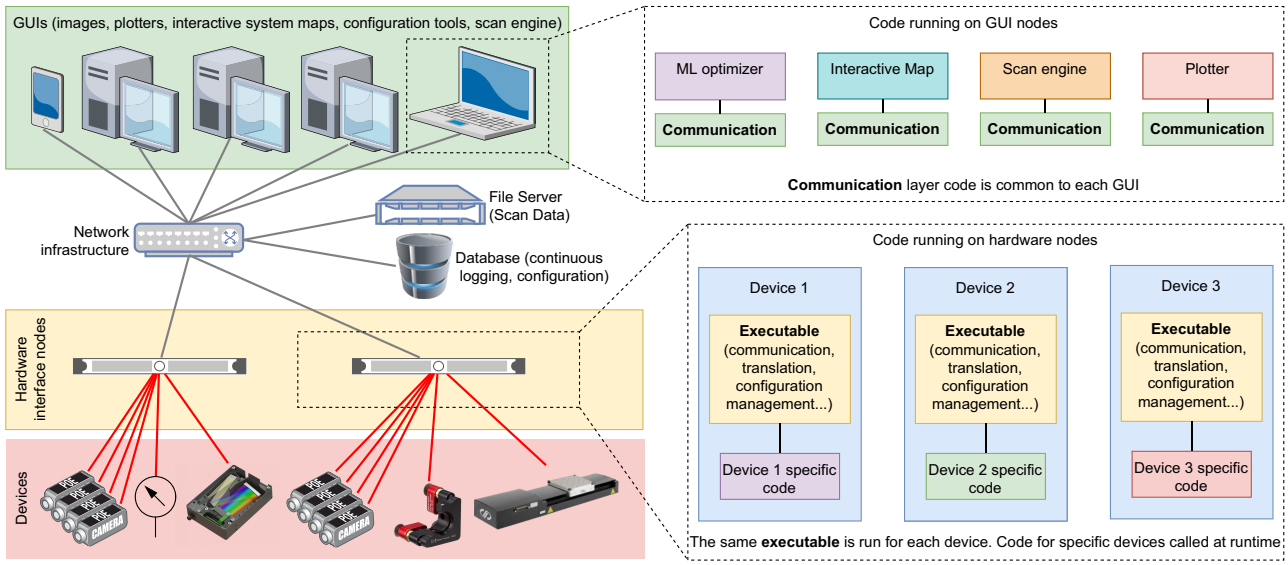

Figure 3. Hardware and software overview of the GEECS control system, highlighting the layered modular approach. The hardware view is shown on the left, with multiple devices attached to a given computer, the number of which is determined by the system resources. The software running on the computers is shown on the right, which sends data and receives commands through the network infrastructure. The GUIs and the database receive data continuously, with the latter limited to scalar data. The file server receives scan data (which includes scalar, vector and image data) when the user initiates a 'scan', which can be linear, from a script, or via the GEECS Python API. All devices are controlled by an instance of the same executable, which calls device-specific code in a plugin architecture. GUIs are custom executables, but share the same communication code to receive data and send commands.

and modular, exhibiting a philosophy similar to well-known platforms such as EPICS, with particular attention to making it easy to install, learn, use, customize and expand. It was developed for the BELLA PW system[68], but has been the control system for all BELLA Center facilities for over a decade.

The DAQ system begins with the hardware devices, such as cameras and motion controllers. GEECS aims to make the job of both the developer and user as easy as possible. For the developer, the software engineering effort for adding a new device type is minimized by limiting the effort to the code specific to that device, eliminating the need to write new user interfaces and configuration mechanisms for each new device type.

GEECS follows the object-oriented programming paradigm, which means that device types or 'classes' inherit from the general device class as well as from each other, minimizing coding effort for adding a new device class. For example, this means the communication systems are automatically implemented for a new device class. Device classes also automatically have common methods, such as 'acquire', 'power off' or 'load configuration'. The general device class is compiled into a single executable 'device.exe'. Any device launched uses this executable file, which then calls any device-specific code required from source distribution folders.

New features can typically be added to individual devices without modifying the shared base executable. For the user, adding a new device is performed through a user interface that communicates with an SQL database describing the experimental configuration (the configuration database). Once a device is added to the database, it can be launched and controlled via a graphical user interface (GUI) named Master Control (MC). A few examples of the features of MC are that it can perform the following: start and shutdown all devices, computers and GUIs associated with an experiment with a single click enabling rapid 'switch-on' (or off) of an experiment; control any device on the system; perform scans of single or multiple variables; save configuration snapshots; and report system alarms. GEECS also includes GUIs for each type of device, which makes configuration and viewing of data convenient. For example, the camera device type GUI allows for image viewing customization (such as colormaps, spatial smoothing and perspective correction) and quick access to typical controls. In addition, users may develop custom GUIs in a few minutes without the need for programming knowledge using LabVIEW and the GEECS publish/subscribe architecture, which is designed to reduce the dependencies between user interfaces and devices as much as possible. It should be noted that although the majority of code in GEECS is written in LabVIEW, Python is increasingly being used to control and monitor GEECS devices, and has proven particularly useful for leveraging various machine learning (ML) tools.

Advantages of using LabVIEW. Compared with traditional programming environments, the LabVIEW programming environment offers a number of advantages, such as reduced programming effort for many tasks, such as interfacing with hardware and taking measurements, the same environment to develop desktop and real-time applications, natural parallel

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

processing, easy field-programmable gate array (FPGA) programming and ease of data visualization. There are disadvantages also, including the need to purchase some of the plugins that reduce coding effort (e.g., NI Vision). The reason why we chose LabVIEW for the BELLA control system in 2009 is that the task of completing a full control system in a lower-level language is significant, and abstraction in LabVIEW allows for rapid generation of code, whether it be the communication layer or hardware interface. This abstraction also allows the control system to quickly adapt to new technologies.

Control system architecture. Although we found certain advantages of using LabVIEW, the GEECS control system architecture is more important than the programming language used. By following standard practices such as ANSI/IEEE-1471-2000, coding effort is minimized and the system becomes far more adaptable. For example, the communication layer currently uses a mix of a custom transmission control protocol (TCP) and user datagram protocol (UDP), but could easily be re-written with abstractions to take advantage of a standardized message-passing library (such as ZeroMQ), since this layer is separate from others. Our choice of object-oriented programming also brings advantages. Adding a new type of camera becomes straightforward since only the code specific to that type of camera needs to be written, while other features are inherited.

Challenges and development path. The biggest challenge encountered has been the limited personnel effort available to develop the control system. Although a critical element to the success of modern LPA experiments, control systems have typically not been a priority. The choice to develop our own control system has allowed us to quickly add new device types with minimal effort, but at the same time it comes with some disadvantages. Although the framework does ensure some level of organization of the code, individual parts (especially the device-specific code) are often rushed to get to a working state and never quite finished or polished. The most critical issue is that the documentation is insufficient. If the main developer ceases to work on this, it would be a challenge for a new developer to take over without help. User documentation is also a challenge, but this is one that can be addressed by encouraging users to contribute. Finally, the lifetime of GEECS is dependent on the continued development and support of LabVIEW by NI.

GEECS for others. Since the framework is based on standard best practices, the philosophy of the control system is similar to that of more widely used platforms. Those platforms could adapt some of the features of GEECS, especially the idea that a fully functional control system can be installed easily and configured quickly with simple GUIs, and that customization and new device classes can be made with little programming knowledge.

3.1.3. Further resources for LabVIEW

The control system from the case study, GEECS, is open source and freely downloadable through the GEECS GitHub page[74]. A link to a GEECS installation guide[75] is also found in this paper's references. The NI Learning Center[76] provides professionally written resources for getting off the ground with LabVIEW. Also, NI has written a whitepaper on getting started with its Distributed Control and Automation Framework (DCAF)[77] (which is not used by GEECS).

3.2. EPICS

3.2.1. Overview of EPICS

EPICS (Experimental Physics and Industrial Control System) is an open-source framework for developing SCADA (supervisory control and data acquisition) systems. Originally developed at Argonne National Laboratory, it is based on applications communicating over the network using named PVs. As well as the core EPICS framework, the community has developed interfaces and drivers to support a wide variety of devices.

EPICS has depth of use and proven rigor in the particle physics, magnetic fusion and astronomy community[78]. EPICS is in use globally by many dozens of major scientific research facilities. Example facilities where EPICS is used today include high-energy-physics beamlines (e.g., Diamond Light Source, SLAC National Accelerator, Advanced Photon Source), astronomy and astrophysics observatories (e.g., W.M. Keck Observatory, LIGO) and magnetic confinement fusion facilities (e.g., KSTAR, ITER). The EPICS collaboration hosts annual meetings and code-athons welcome to anyone using EPICS[79].

One challenge of building EPICS systems for high-power laser facilities is a relatively small number of examples in our own community, especially for university-scale high-power laser facilities. Another challenge for members of our community in adopting EPICS is the need to develop device drivers for those scientific instruments not already in use in the broader EPICS community. Developing new drivers for devices in existing high-power laser systems may be a source of frustration for science teams, especially for small teams who are accustomed to relying on vendor-provided drivers or vendor-provided GUIs to control their scientific instruments. Fortunately, EPICS contains high-level tools to build new drivers and many abstractions that make this work easier (for an example of a versatile device driver for cameras, see areaDetector[80]).

There are often many ways to accomplish the same task in EPICS. For example, there are many alternate tools to display data, many alternate approaches to building data processing pipelines, two official communication protocols, several alternative techniques to remotely manage devices and a variety of pathways to creating device drivers.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

A distributed community of EPICS developers over the past 30 years has created many interacting and overlapping solutions and toolsets. The number of choices among tools, and the deep and decentralized knowledge in the EPICS community, might be considered a long-term feature that also leads to a steep learning curve for new users.

EPICS is designed for control systems and control feedback loops involving PVs (e.g., valves, pumps and pressure sensors), not specifically for scientific DAQ (e.g., high-repetition-rate streak camera data). However, the latest version of EPICS, EPICS v7, includes a new network protocol (pVAccess) with data structures and network performance that are suitable for scientific DAQ[81].

One major institution in the high-power laser community currently using EPICS is the Central Laser Facility (CLF) at Rutherford Appleton Laboratory (RAL). We present their experience with EPICS in the following case study.

3.2.2. *Case study: Central Laser Facility at RAL*

EPICS has been adopted fairly recently by the CLF. It has been incorporated into the control system for the Gemini laser, was used to build the new control systems for HiLase and D100-X and is being used to build the control system for the new Extreme Photonics Application Centre (EPAC) facility[82].

EPICS in Gemini. The Gemini Control System (Figure 4) was originally commissioned in 1997 for what was then the Astra laser facility. Since then, it has undergone a number of upgrades, the most notable being the addition of twin Quantel lasers in 2009 when the facility was renamed Gemini. Since then, the control system has been re-written in .NET, and now uses EPICS.

There are several applications within the control system suite:

(1) the main control system handles the orchestration of triggers, control of various devices and (in conjunction with the interlock safety system) the hand-over of control from one laser area to another;

(2) four target area control systems provide an interface to allow users of the facility to visualize the status of the beamline and to control devices appropriate to them;

(3) a laser area control system allows Gemini operators to visualize the status of the beamline from an operations point-of-view, and to set some of the main operational parameters.

These applications communicate using EPICS PVs. Each PV is a piece of data, usually small, that can be read and sometimes written by other applications. The main control system makes available over 100 PVs. Examples include the laser operating (energy) mode, the status of the wall shutters and a 20-second countdown to the next shot. Applications monitor these PVs for various reasons, including changing device settings depending on the laser energy mode and acquiring data on shot.

Other applications are used to control individual devices in the facility. These also communicate using EPICS PVs, and most of them host their own PVs. For example, the controller for a motorized stage may have a PV that can

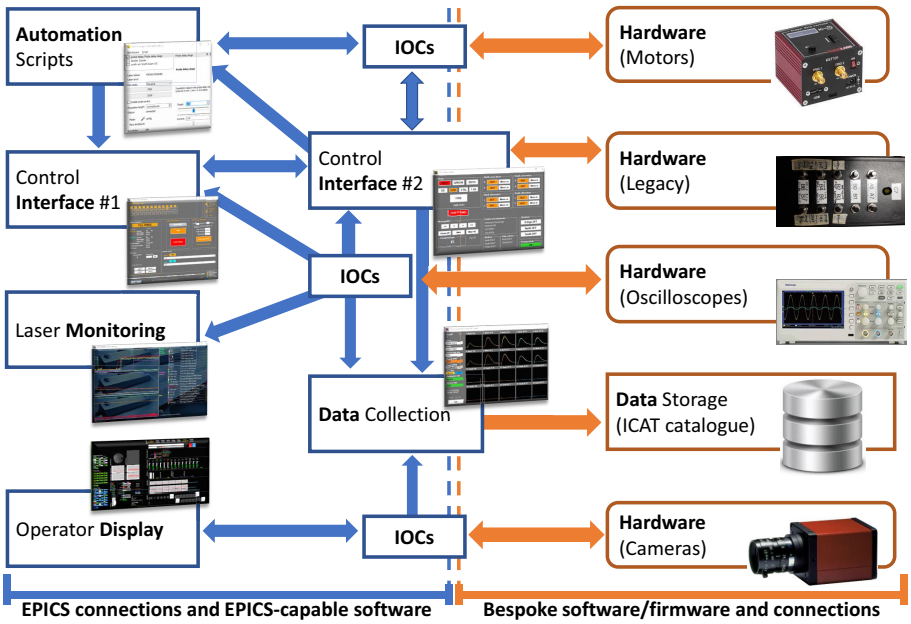

Figure 4. Architecture of the Gemini Control System. EPICS input/output controllers (IOCs) are software + hardware layers providing an abstraction between low-level device hardware and high-level control system software. Legacy hardware is connected into the system through a legacy interface, Control Interface #2. A differentiation is made between EPICS-interfacing components and connections (in blue) and bespoke components and connections (in orange). Clients for device data include human control interfaces, informational laboratory displays, automation software and archival data collectors.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

be written-to to move the stage, and another that can be read-from to find its current position. Controllers for thermometers, pressure gauges, oscilloscopes and other devices all make their data available through EPICS PVs in a similar way.

Any EPICS-aware software on the network can read and write EPICS PVs, so new instruments and applications can be added to the system without disrupting laser operations. Because writing to certain PVs could damage laser systems, Gemini operates within a private network and an EPICS channel access gateway allows read-only access from other parts of the site, operations can be monitored from the office without fear of inadvertent interference.

Diagnostic and metrology data from the laser are collected and stored in a data cataloguing system called ICAT provided by STFC's Scientific Computing Department. The data are then made available to staff and users via a web interface called eCAT, which also provides facilities to filter and analyse the data more fully, to display traces and camera images in detail and to download the data if required.

Experimental automation and DAQ. In contrast to the main operational diagnostics, those in the target area often change with each experiment, so a more flexible DAQ solution is needed. This has not been adapted to use EPICS. The main method used is a custom application called 'Mirage'. Diagnostic software saves data to separate files – one file per laser pulse per diagnostic. Mirage then collects these files and saves them centrally, organized by diagnostic name, by run name and by shot number. This makes it a simple operation to collect all the data for a single laser pulse. Mirage is also able to collect data from EPICS PVs. This is currently limited to environmental data, collected immediately before the laser pulse, but is invaluable for recording experimental settings.

Mirage offers several important features. It displays a summary of which diagnostics are acquiring data and allows users to perform a 'trigger test', to ensure the system is working properly before a laser shot. It can also integrate with other systems. For example, by writing a short Python script, data can be analysed and the results can be plotted in real time.

A more recent feature is the 'experiment automation system'. This allows users or staff members to write Python scripts to control aspects of the experiment. These scripts can move (some) motors, adjust (some) laser parameters and fire laser pulses. Although this system is still considered experimental, it has already been instrumental in some experiments requiring a high degree of automation[38].

EPICS in EPAC. EPAC will build upon the experience and the lessons learned in Gemini while taking advantage of a dedicated team of software engineers to build more robust solutions that are more suitable for scaling to greater levels of complexity and higher data rates. It will also be able to take advantage of the wide variety of drivers developed by the EPICS community, as well as CLF's experience developing EPICS-based control systems from the ground up for HiLase and D100X.

The control system for EPAC has some key differences from Gemini – notably device drivers are mostly implemented with standard EPICS frameworks (such as 'areaDetector' for cameras and 'epics-motor' for motion control). There are also some similarities, with most of the higher-level control system logic being implemented in .NET, which is more suitable for rapid prototyping of complex functionality than the native EPICS alternatives. A comprehensive user interface will be provided for laser operators, with a more limited version for facility users. These will be based on the Blazor framework, opening up the possibility of providing access through a web browser or on tablets and phones. This will be combined with user interfaces built with the widely used Control System Studio.

DAQ and management in EPAC. While DAQ will be performed by EPICS device drivers, a data management system is needed to collect and organize data into HDF5 files, which could be based on NeXus or any other standard format. This system is still under development, with key challenges being the high overall data rate (greater than 1 GB/s expected) and the need to identify which laser pulse each piece of data is associated with.

The key technology for the data management system will be Apache Kafka, a distributed system for handling streams of data. Data will be sent to Kafka either by an EPICS-Kafka forwarder, or by plugins embedded into areaDetector drivers. Any system needing access to the data will be able to access a real-time stream, or recall it during a limited retention period. The latter may be particularly useful after a fault has occurred.

The aim is for a single file to contain all the data for a single sample or scan, as well as all the necessary metadata. Users will be able to access the data through DAaaS (data analysis as a service, see Section 4.3.2), and new systems will be provided for data archiving.

Notably, the greater use of EPICS and standardized interfaces within the EPAC control system should make it simpler for experiment automation systems to control various devices and system parameters. To take advantage of this, we intend to adopt Bluesky, an open-source framework for experiment automation.

Our experience with EPICS. The use of EPICS was a considerable technological leap for the Gemini facility that, until 2018, was based mostly on UDP messaging. Other beamlines within CLF had already started the move; so too had STFC's ISIS, reassured that EPICS was championed by the Diamond Light Source on campus and other large facilities around the world. Like any software system EPICS has its quirks, and we were warned on many occasions that the learning curve was steep. This has proved to be painfully true, but the advantages that it has brought in terms of the

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

flexibility and the ease of integration of new devices and applications have been considerable.

Another advantage is the wide community support. Many of the device drivers needed for EPAC already existed, substantially reducing the amount of custom code that must be written and supported. In addition, EPICS support is available for a wide variety of languages, including Python, MATLAB and LabVIEW, allowing users to create custom applications to interact with the control system.

3.2.3. *Further resources for EPICS*

The EPICS webpage provides a variety of official and community support resources[83] for new members. Notably for members of our community who have never tried this control platform, EPICS can be explored on a virtual machine (VM)[84] to sidestep a lengthy configuration, or integrated into a mockup physical control system built with low-cost Raspberry Pis[85].

3.3. *Tango Controls*

3.3.1. *Overview of Tango Controls*

Tango Controls is a free open-source software toolkit[86] for building object-oriented SCADA systems. It was originally developed at the European Synchrotron Radiation Facility (ESRF)[87] 20 years ago and has now been adopted by many scientific facilities and, in particular, telescopes, accelerators, light sources and associated beamlines around the world. Tango Controls can either be used as the main toolkit for their control system, for a subsystem or together with commercially acquired systems in a local distributed network. Tango Controls relies on an active community[88] of developers and users and is independent of an operating system, supporting a core composed by libraries and API definitions in C++, Java and Python[89].

Tango Controls aims to provide object-oriented programming for distributed heterogeneous systems. The Tango Controls software communication layer is based on Common Object Request Broker Architecture (CORBA)[90]. CORBA brings a standard interface for all the objects and services available using an interface definition language (IDL)[91]. An interoperable object reference (IOR) identifies each object (Figure 5). One of the advantages is that it is not necessary to recompile when adding a new object. Since version 9 of Tango Controls, event-based communications use the zeroMQ library[92].

Before going further in the description of a representative system for the laser–plasma community, let us introduce some language elements specific to Tango. A device is something that needs to be controlled. It can be equipment (e.g., a camera, a motor controller), a set of software functions or an ensemble of equipment (e.g., a deformable mirror and a wavefront sensor). Then we have three tightly

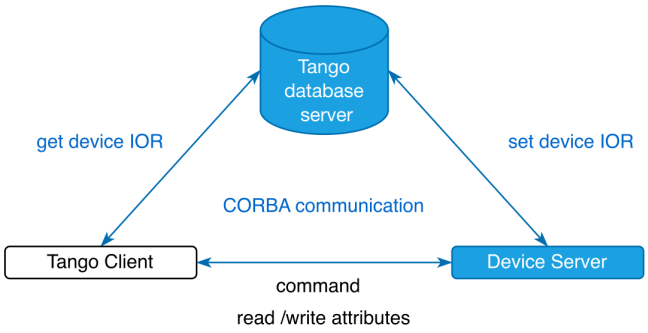

Figure 5. Illustration of the relation between the client and device server and the Tango database running on the TANGO_HOST. This simple system constitutes the minimal Tango configuration.

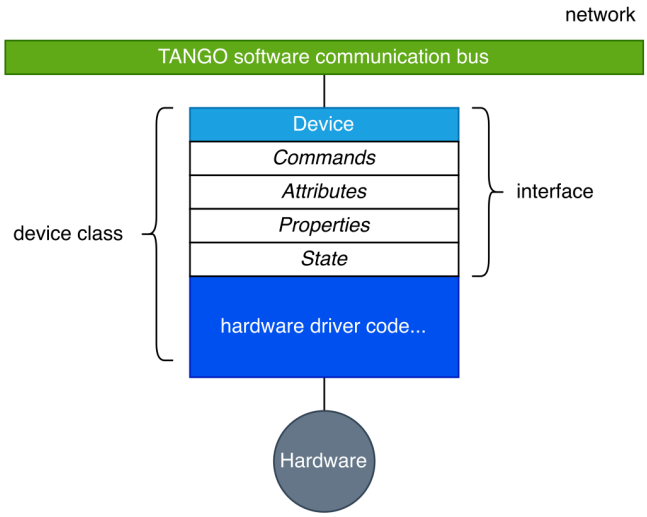

Figure 6. Illustration of the standard Tango device structure between the Tango Controls software communication bus and the hardware. The interface is common to all the devices for all the device classes and can be generated using the POGO tool[93]. The hardware driver part code has to be written by the developer on the top of an existing driver, software development kit or communication protocol.

intertwined concepts. The Tango class defines the interface and the implementation of the device control, as shown in Figure 6: commands, attributes and properties. The commands act on the device (e.g., ON, OFF, RESET), the attributes set/get physical values of the device and properties are the configuration parameters (e.g., IP ADDRESS, PORT NUMBER). The Tango device is an instance of a Tango class giving access to the services of the class. The Tango device server (DS) is the process in which one or more Tango classes are running, each one implementing a device. A hierarchical naming scheme is used for devices. Each device is identified by a fully qualified domain name[94], such as /DOMAIN/FAMILY/MEMBER.

The DS configuration is stored in the Tango database identified as the TANGO_HOST server. The device number and names for a Tango class are defined within the database, while the Tango classes that are part of the DS are defined in

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

the Tango database and in the source code of the DS. The Tango database is associated with a special Tango device performing a centralized storage for the control system configuration parameters and for persistent data. The Tango database is based on a MariaDB database engine. The Tango database is also the service for establishing connections (IOR) between the client and server on the control system. So, the minimum configuration Tango Controls system can be a computer unit running a MariaDB database server, the Tango database device and a DS. On the same computer we can then run a DS for a camera (LIMA[95]), a DS controlling a tip-tilt mirror and a DS being a software function steering a laser beam far field to a given position on the camera.

The device attributes are of various types[86] and support three formats: scalar, spectrum (1D array) and image (2D array). Attributes can be set in three access modes: read, write and read & write. Each device attribute is defined by its properties, which are fixed to five types in which one can find the data information, range value and two essential properties for a distributed control system: alarm and event parameters. When implementing Tango devices in a control system, a hierarchy has to be set starting from the hardware with a low-level device, then a device encompassing the relations between several low-level devices and then higher-level devices with processes and calculations on the sub device attributes data. The communication between DSs is of two main types: the client/server communication, which can be asynchronous or synchronous or event-based with the push/subscribe method. The most used and simple asynchronous type is the polling mode mechanism. It allows the Tango DS to decouple the real device from the client's requests. The Tango DS has specific polling threads that can be configured for polling attributes or commands. The polling results are stored in a buffer with a configurable depth. This implementation helps to monitor the health status of the DS. Since Tango Controls 8, push/subscribe event communication has been available for the attributes. The main categories of pushed events are the change event (absolute or relative change of the attribute that can be configured), a periodic event or an archive event being a mix with the periodic event as the change of value is checked at the polling period. For further details on alarms and logging specific Tango devices, we invite readers to look at the Tango documentation[86].

Several systems in our high-power laser community, particularly European facilities[96], are contributing to the Tango Controls platform, for example, APOLLON[97], ELI-ALPS[98], ELI-BEAMLINES and CALA[99]. Below, we present a case study from one of them: the PALLAS project of the CNRS National Institute for Nuclear and particle physics (IN2P3) hosted at Laboratoire de Physique des 2 infinis Irène Joliot-Curie (IJClab).

3.3.2. Case study: PALLAS

What is the PALLAS project and why is Tango Controls well-suited for this work? The PALLAS (prototyping accelerator based on laser–plasma technology) project hosted at Irène Joliot-Curie Lab (ICJLab) – also known as IJCLab – is developing a laser–plasma injector test facility with the goal of producing electron beams with 200 MeV, 30 pC, less than 5% energy spread and 1 mm·mrad emittance at 10 Hz with comparable stability and reliability to more conventional radio-frequency (RF) accelerators. The project has three main axes of investigation for the laser–plasma-based injector: (1) advanced laser control; (2) development of plasma targetry; and (3) development of a compact electron-beam characterization beamline for studies exploring accelerator staging and beam transport. A state-of-the-art control command and acquisition system is mandatory to achieve the optimizations and systematic studies on the reliability and stability of laser–plasma injectors. Based on the experiences of ICJLab groups on the ThomX project[100] and at neighbouring facilities (ESRF, SOLEIL, APOLLON) who are actively involved in the development of Tango Controls, the choice of Tango Controls has been made for the PALLAS project. A limitation for projects based on laser–plasma acceleration is the diversity of instruments and their use compared to conventional RF particle accelerators, especially for the whole laser-driver control part.

How is Tango Controls integrated into PALLAS? The global architecture of the PALLAS control command and acquisition system is as follows. The laser driver is a customized commercial 40 TW laser system running its own distributed control command system based on the commercial ElliOOs libraries on a separated local network. A Tango gateway between the local distributed network of the laser system and the main PALLAS local network allows the control of pre-selected features of the laser system. The laser features currently accessible are the laser status, main shutter for laser firing, energy, spectrum and beam position and profile at the various stages of the laser system up to the last amplifier. All the hardware of the laser–plasma injector, including the laser transport and compression, is integrated under Tango Controls. The vacuum pumps, gauges, valves and radiation safety system are controlled by a programmable logic controller (PLC) with a Modbus/TCP interface for Tango Controls supervisory control integration. The hardware for the laser-driven plasma accelerator includes motors, area detectors, photo-diode detectors, magnet power supplies, a digitizer and various DAQ cards. The total number of DSs for hardware control is currently 23 without including the DS for archiving operations and accelerator control or optimization.

The laser-driver local network is a 1 Gb/s network with a special link at 10 Gb/s between the main network switch and the data server laserix-arch, while the accelerator network

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

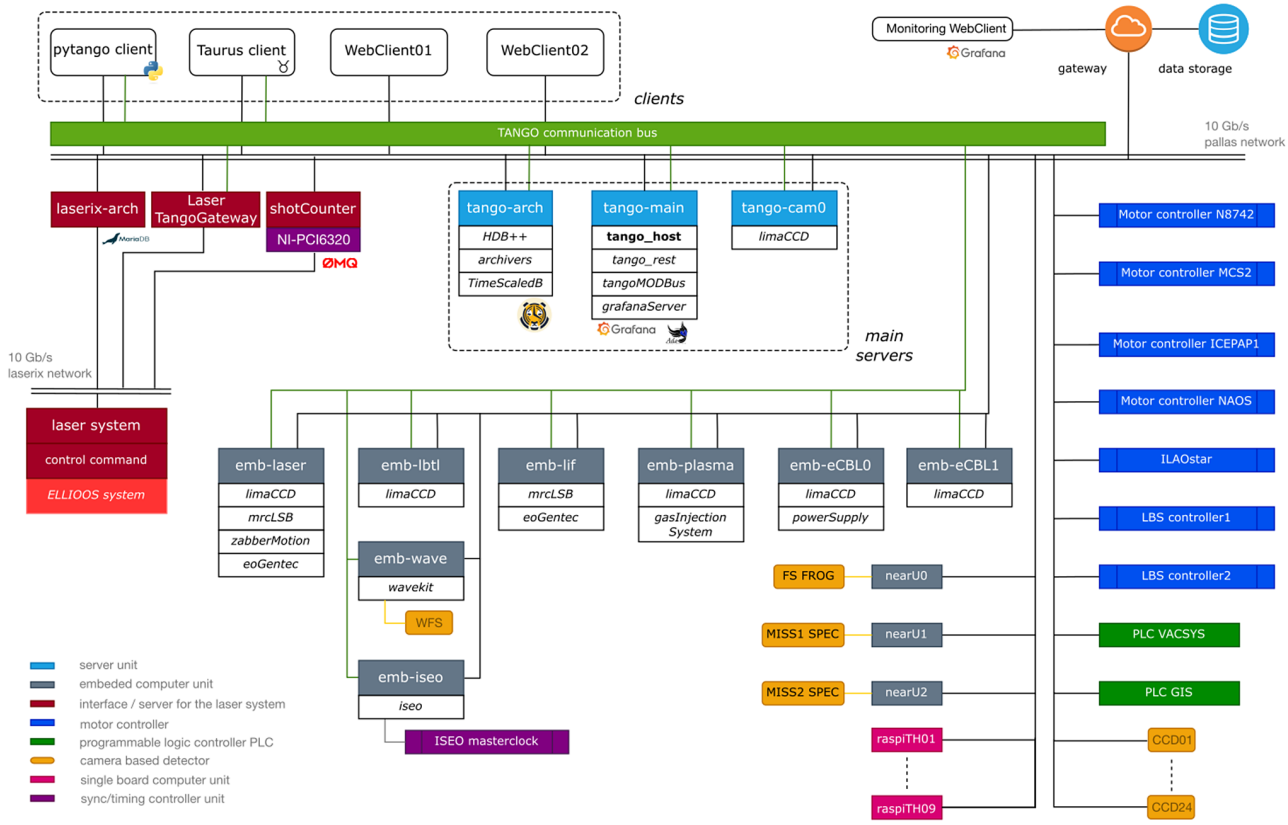

Figure 7. Schematic of the PALLAS control command and acquisition system. System elements are represented by coloured rectangles (colour legend in the lower-left corner), with links between elements represented as coloured lines. Green lines show Tango communication pathways, black lines indicate wired Ethernet connections, yellow lines show wired USB-3 connections and grey lines indicate wired RS232 connections. For better readability, some hardware is not shown. The system can be divided into four parts: (1) the main Tango server units (light blue), the heart of the system running the Tango database, archiving system and webservers; (2) the embedded computer units (light grey) running some Tango device servers (sometimes located adjacent to scientific hardware due to limited cable lengths for USB-3 or RS232 device connections); (3) the hardware motor controllers (dark blue), the PLCs (green) and the area detectors (dark yellow); (4) examples of clients, pyTango and Taurus, which need Tango Controls software installed on a client machine, and WebClients, which requires only a web browser and a connection to the local network. Below the coloured rectangle for each server unit or embedded computer unit, device servers running on that unit are indicated in italic font within white-filled rectangles. These are named LimaCCD, eoGENTENC, mrcLSB… and represent device servers openly developed and available for download[101,102]. The laser system has three connection points into the PALLAS local network: the laserix-arch where laser data are stored, the shotCounter that uniquely identifies laser shots and the TangoGateway running a device server that gives Tango access to certain laser controls. The laser system has its own separated network. As shown in the upper-right corner, some of the webservers can be accessed from either side of the PALLAS local network through a gateway, and data storage is done outside the local network for easier and broader data sharing.

has a speed of 10 Gb/s (see Figure 7). The laser system has its own data logging system based on a MariadB back-end. The event subscriber DSs, or archivers, will be able to gather selected attributes and data from the Tango DS with a unique shot identification and time-stamping given by the 'ShotCounter' system at 10 Hz.

On the accelerator part, the archiving system is currently under development and will be based on the archiving Tango system HDB++[103] with a TimescaledB back-end and the ShotCounter system for event building based on a unique laser shot identification number (64-bit integer) and a configurable laser and system status. The ShotCounter, supplied by the Amplitude Laser, is composed of an embedded PC and an NI PCIe 6320 card[104] with 16 analog and digital I/O associated with a 100 MHz digitization allowed by an NI-STF3[105] counter/timer. The embedded PC of the ShotCounter runs a ZeroMQ[106] server configured in publisher mode that can distribute the event time-stamping and shot identification at up to 100 Hz depending on the triggering configuration of the hardware to both the laser-driver and accelerator local network. The 16 analog inputs of the ShotCounter allow the recording of laser system states (energy level, amplifier injection, opening of the shutter to the accelerator, etc.) by applying a threshold. All the logic is set within the SHOTRECORDER DS, as shown in Figure 8. The recording of attributes can be set depending on the DS communication type and then all the attribute data are pushed by the EVENT SUBSCRIBER DS to the various time series databases.

The data flux is expected to vary between 50 and 250 MB/s depending on the EVENT SUBSCRIBER DS configuration and the number of attributes to be stored. The database architecture is based on a high-availability scheme with write

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

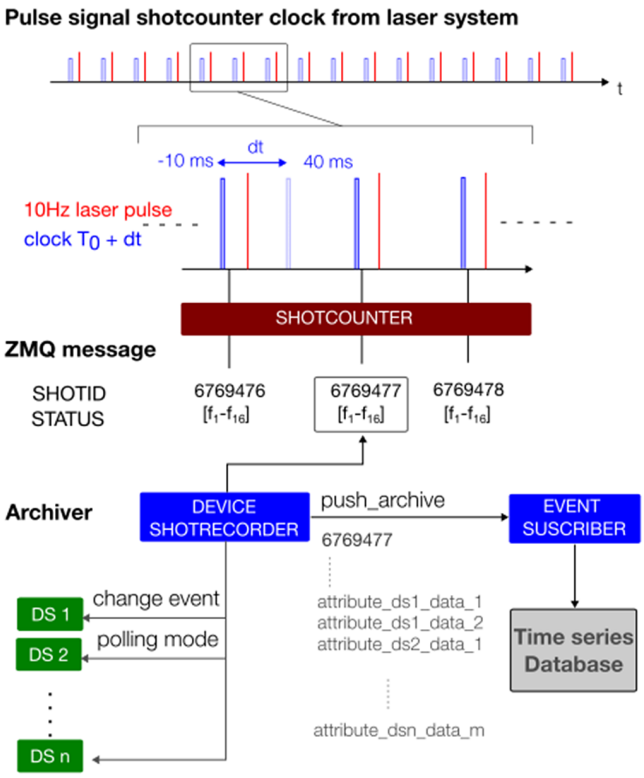

Figure 8. Timing and event generation system for the time and shot stamping of the data. A transistor-transistor logic (TTL) signal can be tuned from −10 to 40 ms around each laser pulse at 10 Hz. The ShotCounter generates a unique 64-bit shot identification. The ZMQ message contains the SHOTID and STATUS of the 16 analog inputs of the NI-PCIe 6320 card. Archiving data is set by the SHOTRECORDER device server (DS) and then an archiver (EVENT SUBSCRIBER DS) stores the data to the target database.

to master database and read to replica for the accelerator beamline and for the laser-driver system.

Tango Controls has a rich graphical client framework. Out of the box tools for Tango Controls include Astor[107] (a graphical tool allowing distant management of DSs without needing to use ssh), Jive[108] (a graphical tool for browsing tango_host database and to configure devices), ATK-Panel[109] (allowing a default or developed GUI to control a device), Pogo (to generate devices), Taurus[110] (to generate and develop a GUI) and Taranta[111] or Waltz[112] (for web GUIs). For the PALLAS project, we foresee leveraging these various graphical clients to suit different needs (such as web interfaces for broad accessibility on mobile and Python for broad integration with other software). First, for the operational user interface of the laser system, laser transport, compression, focal spot characterization and optimization and vacuum system control, we plan to use a cross-platform web-based dashboard with Ada webserver and Plotly libraries[113,114], and an API-REST Tango interface[115]. Second, for scanning and optimization of the laser–plasma injector, Python-based scripts will be developed. Third, for monitoring of the whole system (infrastructure temperature, relative humidity, cooling water flow, gas pressure distribution, main laser-driver parameters, vacuum levels, radiation levels, etc.), we will use Grafana[52] on top of the archiving system. This will give us a distributed dashboard accessible from anywhere, including staff offices and various client posts within the laboratory.

Challenges. One of the challenges for the laser-driven plasma accelerator is the extensive use of large area detectors and the necessity of data reduction/compression before the data is sent to the global network of the Tango Controls system. The laser-driver system uses 20 charge-coupled device (CCD) cameras at 1 and 10 Hz, distributed across two network switches, and two embedded PCs for image processing to determine the laser spot centroid, size and orientation. The accelerator part of the system, including laser transport, optical compression and focusing, as well as full characterization of the laser pulse in the interaction region, includes a total of 25 area detectors for laser beam diagnostics, the electron-beam profile and spectrometers. Most of the laser diagnostics are being integrated into the Tango Controls system with standardization of the client–server configuration with the REST-API protocol between embedded computer units, for example for laser diagnostics such as FROG[116] and 2D spectrometers. This configuration allows for 2D data-array processing and retrieval software in the embedded computer units sending only reduced scalar data and processed image to the Tango DS on request.

3.3.3. Further resources for Tango Controls

Tango Controls provides extensive official documentation[86], an active community forum[88], a searchable archive of a mailing list that preceded the community forum[117] and a Slack channel[118]. Each of these is accessible to new users. Notable for those wishing to quickly try out Tango Controls, a fully configured VM is available[119] for local installation or in cloud server services. Furthermore, the PALLAS project is participating as much as possible in Tango Controls development, favouring an open-source annex API. All the DSs developed to support PALLAS are made available to the community within IN2P3's public GitLab repositories, under 'NOliProg/instrument'[102].

3.4. Intersections and further alternatives

Facility-level approaches to laboratory control can utilize a mix of strategies and technologies while retaining their high-level organization. Planning for a mix of strategies and technologies may particularly benefit the high-power laser science community for two reasons. First, scientists in the high-power laser community often travel and repeat similar experiments at different facilities. When traveling, scientists often bring their own equipment to various facilities. Planning strategies in advance that integrate (rather than isolate) foreign subsystems will strengthen the scientific flexibility

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

of a facility that accommodates traveling scientists. Second, planning for heterogeneity in technologies (rather than rigid uniformity) allows for quick adoption of advances made by other facilities without completely repeating their development effort. For example, EPICS and Tango Controls devices could be graphically controlled by clients running LabVIEW or Taurus[110], or an EPICS client could be patched into the Tango Controls system. Network formats can be translated and adapted between control platforms. Custom Python control code, or low-level C++ instrument drivers, could be interfaced into any of a variety of control platforms. Further alternative control platforms exist, and we have given only a short overview of three that were discussed at a recent workshop on control systems for high-power laser systems[120].

4. Laboratory data management

By coupling the increased access to multi-Hz laser systems with control systems that can maximize the DAQ rate, next-generation experiments will create orders of magnitude more output data than prior generations. Similarly, the explosion in available computing power has enabled greater simulation capability, permitting simulations with higher complexity to be performed within reasonable timescales. In both cases, this data is accompanied by a multitude of metadata that is equally important to the accurate processing and long-term curating of the data. Without the development, or integration, of new approaches to data management and access, the advantages of this increased data rate cannot be realized.

The traditional approach to high-power laser experimental data is very much rooted in 'single shot' experiments. During such experiments, groups of researchers might realistically expect to take a few hundred shots, with some small number of these shots repeated at the same point in the experimental parameter space to provide a measure of stability. The data from all the experimental diagnostics would be stored locally totalling a few tens of Gbits, and copied to portable drives to be transported to different institutes for independent analysis. Within the established approach to simulations, several low-dimension simulations may be run to optimize the simulation setup and scan the parameter space with a limited number of high-cost 'hero' simulations performed at discrete points of interest. Given the low volume of data it is possible for analysis to be strongly customized for individual shots/simulations. In the new systems, with terabytes to petabytes of data, this methodology of independent distributed copies of data, and individually tailored analysis, may not be as effective at leveraging facilities' capabilities in producing the highest impact science. The approach taken to store and interact with data may benefit from adaptation.

The very individual nature of experiments (and simulations) with varying diagnostics, file formats, file structures and analysis requirements also presents a challenge when attempting to combine or compare data between facilities or between different simulation codes. In order to increase efficiency, facilitate data comparison and comply with open access requirements, it is necessary to determine ways in which data can be standardized and usefully accessed by the community. These goals can be advanced by considering the re-use of diagnostic and analysis tools, standardized data formats and cloud-based processing.

In this section, we describe approaches for addressing the challenges of data storage and access.

4.1. Opportunities

4.1.1. Facilitating data access and tooling re-use: F.A.I.R

The term F.A.I.R. is often used together with data and metadata. The acronym stands for findable, accessible, interoperable and reusable[121,122]. In brief, the four letters describe how to enable automated access to and processing of data and metadata.

- **F**indable: use a globally unique and persistent ID for each set, describe data with rich metadata, metadata must include that ID and register the IDs at a searchable source.
- **A**ccessible: access to data and metadata via the ID by using an open and free protocol, also with authentication methods if necessary.
- **I**nteroperable: use a formal/accessible/shared language for knowledge representation, use vocabularies, cross-referencing.
- **R**eusable: provide also attributes for usage license and data provenance, establish and use community standards.

F.A.I.R. data and 'open science' should not be confused despite the usual meaning of the terms 'fair' or 'fairness' referring to humanity and ethical behaviour. F.A.I.R. data rather relates to machine-assisted research methods, and therefore also allows for restricted access in cases where this is demanded by policy. The F.A.I.R. principles basically address the needs for automated data processing but do not prescribe to which extent data and metadata must be publicly and freely available. These aspects are left to be determined by the owner. However, as soon as data accessibility is defined to follow the F.A.I.R. principles – which basically imply machine readability and compliance with a community-wide vocabulary – the step towards open science is relatively small.

A number of the technical requirements listed in the outline above have known solutions. For example, several methods exist to generate globally unique and persistent IDs, for example, DOI, URN or hdl (handle). These systems also register the respective IDs and provide access

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

protocols. Similarly, several languages for knowledge representation exist, including Web Ontology Language[123], ClearTalk and CQL. Yet, the F.A.I.R. principles also suggest that domain-specific, community-wide accepted standards should be jointly defined. This comprises, in contrast to the aforementioned protocols, a consensus across the community about terms and their meaning, that is, a vocabulary of the respective research field. Standardization is therefore an essential step towards F.A.I.R. and such standardization of data, vocabulary and behaviour has been adopted within other research communities, for example, the astrophysics community[124,125].

One significant example of adoption of the F.A.I.R. framework in the high-power laser community is at the Extreme Light Infrastructure (ELI). An ELI data policy document[126] speaks to the rationale and implementation of F.A.I.R. principles in the context of high-power lasers, and can serve as motivation for further adoption in our community. A variety of general F.A.I.R. case studies can also be read online and are linked in our bibliography[127–129].

4.1.2. Definitive data sources and shared tools

One opportunity in having a consistent laboratory approach to data management is to facilitate remote access to original data. Having an accessible, definitive source for original data can be helpful in avoiding mislabelled data or fragmented datasets that might result from passing data from person to person. Furthermore, facilities can provide access to contextual information (such as room temperature and humidity) that typically would not be requested by experimenters, except when hindsight leads to the need.

A data management approach might also involve non-local analysis of the remote datasets using shared analysis tools. Cloud software and cloud environments, configurable and shareable between stakeholders, could leverage distributed data access and high-performance analysis on large datasets that a local computer with a local copy of the data would not. Shared cloud resources for data analysis might lower the barrier to entry for new team members, and strengthen collaboration within and between teams.

Finally, definitive data sources and shared tooling have positive implications for scientific reproducibility. Others can re-trace steps in data analysis more easily, leading to more reproducible scientific analysis. Furthermore, with copious metadata and facility context, researchers can re-trace facility conditions and instrument configurations to reproduce scientific experiments with better fidelity.

4.2. Challenges

Formalizing approaches to data, and increasing accessibility, leads to questions of 'who sets our standards' and 'who has access'.

'Who has access?' When each element of data is treated individually, this question is managed on a case-by-case basis by scientists and facility members. However, with automated structures in place for data management by the facility, data access questions are raised more explicitly and involve the facility to a higher degree. Choices of who does (and does not) have access to data may depend on the scientist preference or facility needs. It may be desired to share certain experimental data to all facility users. Or, it may be desired to limit data to just the facility team, or a single scientific team. One might also decide to limit data access to one group prior to a certain date. Understanding typical science data access patterns among facility stakeholders is important, so that facilities can set up access control tools that are technically capable of enabling these patterns. Best practice is to plan out access control prior to taking data through conversations between the experiment team and the facility team. There is no one solution to access control for all experimental runs.

'Who sets and updates the standards?' In the past, researchers working in isolation have been able to maintain their notes without oversight, leading to huge variations in style and detail. This *modus operandi* is not compatible with transparency and open access, and is especially not compatible with machine processing of metadata. While standards are necessary to support high data loads and open access, there is a new challenge of supporting heterogeneous instruments and data needs within a common framework – and of bringing stakeholders to consensus on this framework.

Existing structure can make certain elements of creating new device data easier, but other elements less flexible. If data standards are too rigid, they can hinder scientific flexibility and may be ignored, weakening the effect of the standards. This particular challenge can be partly mitigated through use of open formats and flexible standards that allow for optional metadata fields and quick new definitions of formats.

4.3. Case studies

We now discuss three concrete examples in our community of tackling the challenges of scientific data management: openPMD, DAaaS and the Michigan Cloud Computing Platform.

4.3.1. OpenPMD, software tools and metadata standards based on F.A.I.R.

As an example for the laser–plasma domain, openPMD[130] is an open meta-standard initially designed for particle-mesh data originating from particle-in-cell (PIC) simulations. Meta-standard means it defines how data shall be stored and organized, where 'storing' includes a description of the data. Hence, in openPMD a number of properties are

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

standardized, such as a self-description and data structuring, while others remain flexible. This has proven to be quite powerful since openPMD has recently been adopted[131] by a number of widely used PIC codes, including PIConGPU, Warp, WarpX, FBPIC, Wake-T, SimEx platform, LUME, ParaTAXIS, OSIRIS, UPIC-Emma, Sirepo/Warp PBA, Sirepo/Warp VND, HiPACE++, ACE3P and CarpetX. Using a common format then allows for common analysis tools that gather in an ever-growing software ecosystem around openPMD. However, while openPMD stores simulation output, it does not store the input. Input files are of course much more dependent on the simulation package; hence, the portability is not required from a technical point-of-view. However, for scientific reasons, inclusion of simulation input – ideally in an abstraction layer – would not only enable comparisons across simulation packages, but would also increase the comprehensibility of the output because it would be set into an intelligible context. This is a goal of the PICMI standard[132].

The situation for experimental data is even more complicated. Many diagnostics fielded in laser–plasma experiments record some kind of 2D data in the form of an image. This can be a laser beam profile or laser focus image, but also an image of a particle beam profile converted via a scintillator. More complex in processing are images of data from spectrometers, either dispersing photons or particles, where the spatial coordinates depend, among other parameters, on the energy of the particle/photon. As of now, openPMD is able to ingest the raw images with the help of its CCD image plugin[133]. Once stored, further openPMD tools can be used. However, detector properties and experiment geometry are crucial for analysis of raw data and must be included for compliance with F.A.I.R. This is the point where further standardization of domain-specific terms comes into play in order to achieve interoperability. The definition of such vocabularies must be done carefully with inclusion of the community to ensure wide validity and acceptance.

In addition, experimental parameters, such as target parameters, are decisive for the interpretation of processed data. A problem emerging hereof is the flow of data and metadata: sometimes, targets are characterized beforehand and often do not exist after the interaction. Broadening this, it can be expensive to automate the annotation of everything of importance that occurs throughout the day in a laboratory. Examples include targets that are characterized entirely outside of the control system, or the unplanned manual adjustment of a mirror. Experiment geometry and detector parameters can be recorded all along an experimental campaign but may change in between shots. Raw data, in contrast, occurs only upon the shots. These are challenges for metadata, and require an experimentalist to help keep the metadata up-to-date. Including these asynchronous changes and DAQs into openPMD would be a significant effort. However, for experimental data from the photon and neutron (PaN) science community there already exists a similar meta-standard, called the NeXus Data Format[134]. It is possible that there might be sufficient analogies in data and metadata structure between laser–plasma experiments and PaN experiments to conceive a F.A.I.R. standard building upon openPMD and NeXus.

4.3.2. Data analysis as a service at the Central Laser Facility

DAaaS was originally developed for users of the ISIS neutron and muon source to analyse the data produced by their experiments. By bringing together experimental data, computing resources and specialized analysis software, it aims to remove the need for users to set up their own analysis environments. By providing everything needed for analysis as well as shared storage, DAaaS reduces the effort needed to add a new collaborator.

DAaaS is made up of VMs running in the STFC cloud. Software is automatically preinstalled, and the VMs have access to shared storage containing experimental data. Access to this shared storage is controlled, so that a user can only see data for experiments in which they are involved. Users log into the system through a web browser and select which kind of 'workspace' they require. This determines the computing resources (CPU, GPU, RAM) available to the VM, as well as the preinstalled analysis software. The system then obtains a freshly installed VM from a pool. The user can then access the desktop of the VM (the XFCE desktop environment, running on Linux) through their web browser.

The VM provides access to a per-user persistent home directory, as well as read-only storage for experimental data and a read–write shared drive for analysis scripts and outputs. This allows users to collaborate with other users on the same experiment, by sharing their analysis work.

DAaaS is managed by STFC's Scientific Computing Department, and has now been rolled out across the CLF. Experimental data can be uploaded to the system through dedicated file servers. In Gemini, this can be done automatically by Mirage. This allows DAaaS to be used to analyse data immediately after it has been acquired.

Since being introduced, DAaaS has become popular among the CLF user community and has been used to build ML models of experimental data[135], highlighting its suitability for data management and analysis.

4.3.3. Michigan Cloud Computing Platform for data management

The Michigan Cloud Computing Platform is a unified cloud platform for data management from numerical or physical experiments being developed by researchers at the University of Michigan. The platform seeks to enable F.A.I.R. principles of data management, as discussed previously, for small academic research teams and, therefore, to greatly lower the barrier to entry for performing ML on collected data.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

The platform is built on MLFlow[136], an open-source software developed for managing ML experiments, and the infrastructure-as-a-service tools provided by the public cloud vendors for example, Amazon Web Services. There are two distinct usage patterns to the platform that cover the initial data gathering and subsequent analysis, as follows.

- **Usage pattern 1** – at data collection time, the researcher is able to use a secure API to communicate (physical or numerical) experimental data to the server, which then organizes the data within the remote database and object storage.
- **Usage pattern 2** – data can be accessed, analysed and downloaded securely via the web using the built-in GUI within MLFlow. The platform also has the ability to launch Jupyter Lab environments with flexible computing power (4–192 CPUs, 4 GB–1 TB of RAM) within the secured virtual private network. This enables researchers to perform data transformations and visualization from anywhere where they can access the web.

We find that for our use case of storing, with redundancy, simulation and experimental data for a small academic group that amounts to approximately 25 TB, the storage costs are approximately 500 USD/month, and the server and database cost approximately 200 USD/month. If cloud data storage costs are prohibitive, MLFlow can also hook into other artifact stores (see https://mlflow.org/docs/latest/tracking.html#artifact-stores) such as a typical network file system (NFS) mount, or even a Hadoop Distributed File System.

Advantages of MLFlow. The difference between local data management and cloud-based data management is primarily in the amount of work needed to build, deploy and maintain the infrastructure, as well as in the ability to distribute access across varying geographic regions. The primary infrastructures of any data management platform are servers, databases and object storage mechanisms, as well as the networking and security layers.

The public cloud providers have infrastructure-as-a-service capabilities for networking, security, servers, databases and object storage. This takes the 'undifferentiated heavy-lifting' of building, deploying, securing and maintaining these constructs away from the scientific group.

Now, the scientific group is primarily charged with customizing the data structure and adapting their existing workflow. Using a pay-as-you-go approach also provides the scientific group with the freedom to upgrade/downgrade their hardware requirements as they wish. This helps mitigate possible restrictions from previous purchases and contracts that lock research groups into specific toolkits and hardware constraints. In addition, it enables the scientific research group to be able to leverage the latest technology, for example, GPU accelerators for training neural networks.

Access to these services is provided through well-developed, performant and stable software that is managed by the cloud companies. In fact, there exists a whole ecosystem of companies vying for market-share in this space alone. These well-developed software tools ensure the scientific group's accessibility to the data stored in the cloud. In a final note on data accessibility, the public cloud vendors also invest heavily in redundancy in their systems. In fact, they advertise that their systems, when configured properly, only lose data very infrequently (better than one part in a billion) and rarely have outages in service.

It is also important to consider the data security needs of an academic research group. Using a cloud-based approach enables the research group to leverage the significant investments made by the cloud companies in order to court high-data-security industries, such as the financial services industry. This ensures that the data-security needs of a small–medium-scale scientific research group are met with relatively little overhead by simply just using the standard workflows and permissioning schemes prescribed by the cloud provider. Data, by default, is not publicly accessible, but rather, is made available to access by the person or team given the authority to manage the permissions.

It is likely that larger labs and companies that use cloud-based-solutions for (computational or physical) scientific data management have internal solutions that do something similar to our proposed solution but broader in scope. However, at this time, it is not clear to us whether there are easily accessible, alternative cloud solutions for scientific researchers that can be maintained by one or two cloud engineers. This is why we propose our solution. The primary advantages are that it is with relatively low-budget, is easy-to-implement and maintain in an academic research setting and can be scaled up or down, resource-wise, as is necessary.

A clear disadvantage is using MLFlow as a layer between the user and the database. This is not the most performant implementation but is much more user-friendly. As the data management effort scales up and teams can be tasked with maintaining the platform, it is likely that a more performant and scalable implementation does not have the MLFlow layer as an intermediary and uses an alternative for accessibility, such as a GUI.

Challenges of MLFlow. The primary challenge we have encountered so far is to convince researchers to adapt their existing workflow and use the platform in order to derive benefits from it over the medium to long term.

Historically, the typical researcher was incentivized to extract what they aimed for out of the data and discard the data. While the platform discussed here decreases the friction of collecting and using data over a medium-to-long term, it still remains a challenge to incentivize researchers and decrease the friction with regards to adoption in the short term.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

This challenge is partially addressed by the draw of big data and ML. As the high-power laser community develops novel methods and performs research using ML, the data management strategy for each individual research group becomes increasingly important. This is because ML requires data management principles in two flavours.

First, and most obviously, running the ML algorithms, at scale, on a dataset requires the dataset to be organized, cleaned and easily accessible. This is a primary principle behind the development of this platform.

The second flavour of data management required for ML comes from a more subtle point. Each instantiation of the ML algorithm over a dataset can be considered a numerical experiment. In this case, much like with a simulation, it is also important to track the different parameters provided to the ML experiments and be able to easily correlate those to the outcome of the numerical experiment.

For these two reasons, the draw of developing ML techniques in high-power laser research provides an incentive to the small-group researcher to improve data management principles in the short term.

At small-to-medium-scale facilities, the people running the facility may find it useful to collect data over many experiments, months and years in order to better understand the performance and behaviour over time. This can also drive an interest in having a platform that can collect and organize this data.

Another challenge is having the right expertise in place to be able to deploy such a platform for a small academic research group or facility. While cloud platforms offer a GUI that can help launch the right servers, networking tools, databases, etc., it can be quite cumbersome to repeat this task as well as debug it if something goes awry. To address this, we have developed the platform using an infrastructure-as-code approach. This effectively means that there is a codebase that, when run, deploys the entire platform readily into different cloud environments, for example, each belonging to a different team or group. This reduces the time required to replicate the deployment of the platform from days to minutes.

Community input. One of the central beliefs behind developing our platform is that over the lifetime of a (numerical or physical) experiment, data generated by researchers can often get 'siloed'. With the growing applications of algorithms that can provide data-driven insights, the 'siloed' data represents potentially useful information that is effectively lost.

To be able to best utilize these data in downstream, ML-related tasks, it would behove researchers to approach data collection and management as a longer-term effort than just the short time spent performing and analysing the numerical or physical experiment. Our platform hopefully lowers the barrier to entry for such an effort.

We are currently using the platform as a test-bed for managing experimental data for a user facility, for managing simulation and ML data within an academic research group[137] and for ML experiment management.

How can I try MLFlow? Most current open-source ML experiment managers can run locally on a personal computer where the object and database storage are also hosted locally. This configuration can be installed as a Python package in the typical manner (using the Python package management software 'pip') and executed from the command line. However, the experiment manager system can also be deployed in a more scalable manner onto a remote server with long-term object storage and a database. In this case, it is preferable to build a Docker container to deploy on the remote server. At the time of this writing, we prefer using MLFlow as our experiment manager, but there are many other similar open-source experiment managers that serve a similar purpose.

5. Conclusion

The high-power laser community is at a critical moment in laying down new, long-lasting digital infrastructure for its facilities. Through this manuscript, which addresses a pressing community need, we hope to facilitate open and efficient future collaboration. We aim to spearhead this by supporting knowledge-sharing of the work being done to develop both control systems and data management to meet the needs of the next generation of high-power laser experiments. A distributed networked control system can facilitate laboratory-wide operational speeds and closed-loop approaches, which humans cannot achieve. A consistent approach to managing data can increase data accessibility to scientists and external partners, increase the reliability of metadata and increase the re-usability of data analysis software. In this manuscript, we explored considerations for practical facility-level decision making in these areas, and we highlighted several specific control systems and approaches to data management from our community. By taking steps now to communicate and synchronize, our community can access the benefits, and mitigate the challenges, of our next-generation-facility digital infrastructure.

Acknowledgements

A.J. acknowledges the support from DOE Grant # DE-SC0016804.

References

1. S. Corde, K. T. Phuoc, G. Lambert, R. Fitour, V. Malka, A. Rousse, A. Beck, and E. Lefebvre, Rev. Mod. Phys. **85**, 1 (2013).
2. J. P. Couperus, R. Pausch, A. Köhler, O. Zarini, J. M. Krämer, M. Garten, A. Huebl, R. Gebhardt, U. Helbig, S. Bock, K. Zeil, A. Debus, M. Bussmann, U. Schramm, and A. Irman, Nat. Commun. **8**, 487 (2017).

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

3. J. Götzfried, A. Döpp, M. F. Gilljohann, F. M. Foerster, H. Ding, S. Schindler, G. Schilling, A. Buck, L. Veisz, and S. Karsch, Phys. Rev. X **10**, 041015 (2020).
4. F. M. Foerster, A. Döpp, F. Haberstroh, K. V. Grafenstein, D. Campbell, Y.-Y. Chang, S. Corde, J. P. C. Cabadağ, A. Debus, M. F. Gilljohann, A. F. Habib, T. Heinemann, B. Hidding, A. Irman, F. Irshad, A. Knetsch, O. Kononenko, A. M. de la Ossa, A. Nutter, R. Pausch, G. Schilling, A. Schletter, S. Schöbel, U. Schramm, E. Travac, P. Ufer, and S. Karsch, Phys. Rev. X **12**, 041016 (2022).
5. W. P. Leemans, A. J. Gonsalves, H.-S. Mao, K. Nakamura, C. Benedetti, C. B. Schroeder, C. Tóth, J. Daniels, D. E. Mittelberger, S. S. Bulanov, J.-L. Vay, C. G. R. Geddes, and E. Esarey, Phys. Rev. Lett. **113**, 245002 (2014).
6. A. J. Gonsalves, K. Nakamura, J. Daniels, C. Benedetti, C. Pieronek, T. C. H. de Raadt, S. Steinke, J. H. Bin, S. S. Bulanov, J. van Tilborg, C. G. R. Geddes, C. B. Schroeder, C. Tóth, E. Esarey, K. Swanson, L. Fan-Chiang, G. Bagdasarov, N. Bobrova, V. Gasilov, G. Korn, P. Sasorov, and W. P. Leemans, Phys. Rev. Lett. **122**, 084801 (2019).
7. S. Kneip, S. R. Nagel, S. F. Martins, S. P. D. Mangles, C. Bellei, O. Chekhlov, R. J. Clarke, N. Delerue, E. J. Divall, G. Doucas, K. Ertel, F. Fiuza, R. Fonseca, P. Foster, S. J. Hawkes, C. J. Hooker, K. Krushelnick, W. B. Mori, C. A. J. Palmer, K. T. Phuoc, P. P. Rajeev, J. Schreiber, M. J. V. Streeter, D. Urner, J. Vieira, L. O. Silva, and Z. Najmudin, Phys. Rev. Lett. **103**, 035002 (2009).
8. H. T. Kim, V. B. Pathak, C. I. Hojbota, M. Mirzaie, K. H. Pae, C. M. Kim, J. W. Yoon, J. H. Sung, and S. K. Lee, Appl. Sci. **11**, 5831 (2021).
9. J. Badziak, J. Phys. Conf. Ser. **959**, 012001 (2018).
10. J. Schreiber, P. R. Bolton, and K. Parodi, Rev. Sci. Instrum. **87**, 071101 (2016).
11. A. Alejo, H. Ahmed, A. Green, S. R. Mirfayzi, M. Borghesi, and S. Kar, Nuovo Cimento C **38**, 188 (2015).
12. F. Albert, A. G. R. Thomas, S. P. D. Mangles, S. Banerjee, S. Corde, A. Flacco, M. Litos, D. Neely, J. Vieira, Z. Najmudin, R. Bingham, C. Joshi, and T. Katsouleas, Plasma Phys. Controll. Fusion **56**, 084015 (2014).
13. J. Wenz, S. Schleede, K. Khrennikov, M. Bech, P. Thibault, M. Heigoldt, F. Pfeiffer, and S. Karsch, Nat. Commun. **6**, 7568 (2015).
14. J. M. Cole, J. C. Wood, N. C. Lopes, K. Poder, R. L. Abel, S. Alatabi, J. S. J. Bryant, A. Jin, S. Kneip, K. Mecseki, D. R. Symes, S. P. D. Mangles, and Z. Najmudin, Sci. Rep. **5**, 13244 (2015).
15. A. Döpp, L. Hehn, J. Götzfried, J. Wenz, M. Gilljohann, H. Ding, S. Schindler, F. Pfeiffer, and S. Karsch, Optica **5**, 199 (2018).
16. W. Wang, K. Feng, L. Ke, C. Yu, Y. Xu, R. Qi, Y. Chen, Z. Qin, Z. Zhang, M. Fang, J. Liu, K. Jiang, H. Wang, C. Wang, X. Yang, F. Wu, Y. Leng, J. Liu, R. Li, and Z. Xu, Nature **595**, 516 (2021).
17. R. Pompili, D. Alesini, M. P. Anania, S. Arjmand, M. Behtouei, M. Bellaveglia, A. Biagioni, B. Buonomo, F. Cardelli, M. Carpanese, E. Chiadroni, A. Cianchi, G. Costa, A. Del Dotto, M. Del Giorno, F. Dipace, A. Doria, F. Filippi, M. Galletti, L. Giannessi, A. Giribono, P. Iovine, V. Lollo, A. Mostacci, F. Nguyen, M. Opromolla, E. Di Palma, L. Pellegrino, A. Petralia, V. Petrillo, L. Piersanti, G. Di Pirro, S. Romeo, A. R. Rossi, J. Scifo, A. Selce, V. Shpakov, A. Stella, C. Vaccarezza, F. Villa, A. Zigler, and M. Ferrario, Nature **605**, 659 (2022).
18. O. Graydon, Nat. Photonics **16**, 750 (2022).
19. N. Esplen, M. S. Mendonca, and M. Bazalova-Carter, Phys. Med. Biol. **65**, 23TR03 (2020).
20. P. Chaudhary, G. Milluzzo, H. Ahmed, B. Odlozilik, A. McMurray, K. M. Prise, and M. Borghesi, Front. Phys. **9**, 75 (2021).
21. J. Zhang, H. Jiao, B. Ma, Z. Wang, and X. Cheng, Opt. Eng. **57**, 121909 (2018).
22. Z. Li, Y. Leng, and R. Li, Laser Photonics Rev. **17**, 2100705 (2022).
23. I. Prencipe, J. Fuchs, S. Pascarelli, D. W. Schumacher, R. B. Stephens, N. B. Alexander, R. Briggs, M. Büscher, M. O. Cernaianu, A. Choukourov, M. De Marco, A. Erbe, J. Fassbender, G. Fiquet, P. Fitzsimmons, C. Gheorghiu, J. Hund, L. G. Huang, M. Harmand, N. J. Hartley, A. Irman, T. Kluge, Z. Konopkova, S. Kraft, D. Kraus, V. Leca, D. Margarone, J. Metzkes, K. Nagai, W. Nazarov, P. Lutoslawski, D. Papp, M. Passoni, A. Pelka, J. P. Perin, J. Schulz, M. Smid, C. Spindloe, S. Steinke, R. Torchio, C. Vass, T. Wiste, R. Zaffino, K. Zeil, T. Tschentscher, U. Schramm, and T. E. Cowan, High Power Laser Sci. Eng. **5**, e17 (2017).
24. T. Chagovets, S. Stanček, L. Giuffrida, A. Velyhan, M. Tryus, F. Grepl, V. Istokskaia, V. Kantarelou, T. Wiste, J. C. H. Martin, F. Schillaci, and D. Margarone, Appl. Sci. **11**, 1680 (2021).
25. K. M. George, J. T. Morrison, S. Feister, G. K. Ngirmang, J. R. Smith, A. J. Klim, J. Snyder, D. Austin, W. Erbsen, K. D. Frische, J. Nees, C. Orban, E. A. Chowdhury, and W. M. Roquemore, High Power Laser Sci. Eng. **7**, e50 (2019).
26. F. Treffert, G. D. Glenn, H.-G. J. Chou, C. Crissman, C. B. Curry, D. P. DePonte, F. Fiuza, N. J. Hartley, B. Ofori-Okai, M. Roth, S. H. Glenzer, and M. Gauthier, Phys. Plasmas **29**, 123105 (2022).
27. S. D. Kraft, L. Obst, J. Metzkes-Ng, H.-P. Schlenvoigt, K. Zeil, S. Michaux, D. Chatain, J.-P. Perin, S. N. Chen, J. Fuchs, M. Gauthier, T. E. Cowan, and U. Schramm, Plasma Phys. Controll. Fusion **60**, 044010 (2018).
28. P. Puyuelo-Valdes, J.-L. Henares, F. Hannachi, T. Ceccotti, J. Domange, M. Ehret, E. D'Humieres, L. Lancia, J.-R. Marquès, J. Santos, and M. Tarisien, Proc. SPIE **11037**, 110370B (2019).
29. J. A. Oertel, C. W. Barnes, M. Demkowicz, G. Dyer, M. Farrell, M. Green, R. Muenchausen, A. Nikroo, and I. Prencipe, "Adaptive sample preparation and target fabrication for high-throughput materials science," Technical Report, No. LA-UR-19-26624 (Los Alamos National Laboratory, 2019).
30. E. S. Grace, T. Ma, Z. Guang, R. A. Simpson, G. G. Scott, D. Mariscal, B. Stuart, and R. Trebino, Plasma Phys. Controll. Fusion **63**, 124005 (2021).
31. L. Obst, J. Metzkes-Ng, S. Bock, G. E. Cochran, T. E. Cowan, T. Oksenhendler, P. L. Poole, I. Prencipe, M. Rehwald, C. Rödel, H.-P. Schlenvoigt, U. Schramm, D. W. Schumacher, T. Ziegler, and K. Zeil, Plasma Phys. Controll. Fusion **60**, 054007 (2018).
32. L. J. Waxer, C. Dorrer, A. Kalb, E. M. Hill, and W. Bittle, Proc. SPIE **10522**, 105221E (2018).
33. M. C. Downer, R. Zgadzaj, A. Debus, U. Schramm, and M. C. Kaluza, Rev. Mod. Phys. **90**, 035002 (2018).
34. P. W. Hatfield, J. A. Gaffney, G. J. Anderson, S. Ali, L. Antonelli, S. B. du Pree, J. Citrin, M. Fajardo, P. Knapp, B. Kettle, B. Kustowski, M. J. MacDonald, D. Mariscal, M. E. Martin, T. Nagayama, C. A. J. Palmer, J. L. Peterson, S. Rose, J. J. Ruby, C. Shneider, M. J. V. Streeter, W. Trickey, and B. Williams, Nature **593**, 351 (2021).
35. A. Döpp, C. Eberle, S. Howard, F. Irshad, J. Lin, and M. Streeter, arXiv:2212.00026 (2022).
36. B. Loughran, M. J. V. Streeter, H. Ahmed, S. Astbury, M. Balcazar, M. Borghesi, N. Bourgeois, C. B. Curry, S. J. D. Dann, S. DiIorio, N. P. Dover, T. Dzelzanis, O. C. Ettlinger, M. Gauthier, L. Giuffrida, G. D. Glenn, S. H. Glenzer, J.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

S. Green, R. J. Gray, G. S. Hicks, C. Hyland, V. Istokskaia, M. King, D. Margarone, O. McCusker, P. McKenna, Z. Najmudin, C. Parisuaña, P. Parsons, C. Spindloe, D. R. Symes, A. G. R. Thomas, F. Treffert, N. Xu, and C. J. Palmer, High Power Laser Sci. Eng. **11**, e35 (2023).
37. S. Jalas, M. Kirchen, P. Messner, P. Winkler, L. Hübner, J. Dirkwinkel, M. Schnepp, R. Lehe, and A. R. Maier, Phys. Rev. Lett. **126**, 104801 (2021).
38. R. J. Shalloo, S. J. D. Dann, J.-N. Gruse, C. I. D. Underwood, A. F. Antoine, C. Arran, M. Backhouse, C. D. Baird, M. D. Balcazar, N. Bourgeois, J. A. Cardarelli, P. Hatfield, J. Kang, K. Krushelnick, S. P. D. Mangles, C. D. Murphy, N. Lu, J. Osterhoff, K. Põder, P. P. Rajeev, C. P. Ridgers, S. Rozario, M. P. Selwood, A. J. Shahani, D. R. Symes, A. G. R. Thomas, C. Thornton, Z. Najmudin, and M. J. V. Streeter, Nat. Commun. **11**, 6355 (2020).
39. S. J. D. Dann, C. D. Baird, N. Bourgeois, O. Chekhlov, S. Eardley, C. D. Gregory, J.-N. Gruse, J. Hah, D. Hazra, S. J. Hawkes, C. J. Hooker, K. Krushelnick, S. P. D. Mangles, V. A. Marshall, C. D. Murphy, Z. Najmudin, J. A. Nees, J. Osterhoff, B. Parry, P. Pourmoussavi, S. V. Rahul, P. P. Rajeev, S. Rozario, J. D. E. Scott, R. A. Smith, E. Springate, Y. Tang, S. Tata, A. G. R. Thomas, C. Thornton, D. R. Symes, and M. J. V. Streeter, Phys. Rev. Accel. Beams **22**, 041303 (2019).
40. C. Spindloe, Y. Fukuda, P. Fitzsimmons, K. Du, and C. Danson, High Power Laser Sci. Eng. **6**, e13 (2018).
41. P. V. Heuer, S. Feister, D. B. Schaeffer, and H. G. Rinderknecht, Phys. Plasmas **29**, 110401 (2022).
42. X. Ge, F. Yang, and Q.-L. Han, Inform. Sci. **380**, 117 (2017).
43. U. Schramm, M. Bussmann, A. Irman, M. Siebold, K. Zeil, D. Albach, C. Bernert, S. Bock, F. Brack, J. Branco, J. P. Couperus, T. E. Cowan, A. Debus, C. Eisenmann, M. Garten, R. Gebhardt, S. Grams, U. Helbig, A. Huebl, T. Kluge, A. Köhler, J. M. Krämer, S. Kraft, F. Kroll, M. Kuntzsch, U. Lehnert, M. Loeser, J. Metzkes, P. Michel, L. Obst, R. Pausch, M. Rehwald, R. Sauerbrey, H. P. Schlenvoigt, K. Steiniger, and O. Zarini, J. Phys.: Conf. Ser. **874**, 012028 (2017).
44. D. Albach, M. Loeser, M. Siebold, and U. Schramm, High Power Laser Sci. Eng. **7**, e1 (2019).
45. M. Siebold, F. Roeser, M. Loeser, D. Albach, and U. Schramm, Proc. SPIE **8780**, 878005 (2013).
46. P. D. Michel, J. Large-Scale Res. Facilit. **2**, A39 (2016).
47. https://invigon.de/en/products/camera-software.
48. https://freefilesync.org.
49. https://www.mediawiki.org.
50. https://scicatproject.github.io.
51. https://www.mongodb.com.
52. https://grafana.com/oss/grafana.
53. P. Jenni, M. Nessi, M. Nordberg, and K. Smith, "ATLAS high-level trigger, data-acquisition and controls," Technical design report (ATLAS Collaboration, 2003).
54. N. Berger, T. Bold, T. Eifert, G. Fischer, S. George, J. Haller, A. Hoecker, J. Masik, M. Z. Nedden, V. P. Reale, C. Risler, C. Schiavi, J. Stelzer, and X. Wu, J. Phys.: Conf. Ser. **119**, 022013 (2008).
55. The ATLAS collaboration, J. Instrum. **15**, P10004 (2010).
56. J. B. Thayer, G. Carini, W. Kroeger, C. O'Grady, A. Perazzo, M. Shankar, and M. Weaver, in 2017 *IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)* (IEEE, 2017), p. 1.
57. S. J. Gessner, in *10th International Beam Instrumentation Conference* (JACoW Publishing, 2021), p. 446.
58. W. B. Rouse, *People and Organizations: Explorations of human-centered design* (John Wiley & Sons, 2007).
59. A. Hidvegi, P. Gessler, K. Rehlich, and C. Bohm, IEEE Trans. Nucl. Sci. **58**, 1852 (2011).
60. A. Gaget, *MRF Timing System Design at SARAF* (JACoW Publishing, 2022)
61. K. S. White, K. A. Brown, P. S. Dyer, and V. R. W. Schaa, *Proceedings of the 17th International Conference on Accelerator and Large Experimental Physics Control Systems: Timing and Synchronization* (JACoW Publishing, 2019). https://accelconf.web.cern.ch/icalepcs2019/html/clas013.htm.
62. K. Furukawa, Y. Yan, Y. Leng, Z. Chen, and V. Schaa, *Proceedings of the 18th International Conference on Accelerator and Large Experimental Physics Control Systems: Timing Systems, Synchronization and Real-Time Applications* (JACoW Publishing, 2021). https://accelconf.web.cern.ch/icalepcs2021/html/clas014.htm.
63. Texas Instruments, https://www.ti.com/lit/an/snla098a/snla098a.pdf.
64. https://people.cs.rutgers.edu/pxk/417/notes/ptp.html.
65. http://wiki.zeromq.org.
66. http://www.mrf.fi/index.php/product-price-list.
67. https://www.microsemi.com/product-directory/3425-timing-synchronization.
68. K. Nakamura, H.-S. Mao, A. J. Gonsalves, H. Vincenti, D. E. Mittelberger, J. Daniels, A. Magana, C. Toth, and W. P. Leemans, IEEE J. Quantum Electron. **53**, 1200121 (2017).
69. S. Hakimi, L. Obst-Huebl, A. Huebl, K. Nakamura, S. S. Bulanov, S. Steinke, W. P. Leemans, Z. Kober, T. M. Ostermayr, T. Schenkel, A. J. Gonsalves, J.-L. Vay, J. van Tilborg, C. Toth, C. B. Schroeder, E. Esarey, and C. G. R. Geddes, Phys. Plasmas **29**, 083102 (2022).
70. F. Isono, J. van Tilborg, S. K. Barber, J. Natal, C. Berger, H.-E. Tsai, T. Ostermayr, A. Gonsalves, C. Geddes, and E. Esarey, High Power Laser Sci. Eng **9**, e25 (2021).
71. H.-E. Tsai, C. G. R. Geddes, T. Ostermavr, G. O. Muñoz, J. van Tilborg, S. K. Barber, F. Isono, H.-S. Mao, K. K. Swanson, R. Lehe, A. J. Gonsalves, K. Nakamura, C. Toth, C. B. Schroeder, E. Esarey, and W. P. Leemans, In *2018 IEEE Advanced Accelerator Concepts Workshop (AAC)* (IEEE, 2018), p. 1.
72. IEEE, *1471-2000 - IEEE Recommended Practice for Architectural Description for Software-Intensive Systems* (IEEE, 2000).
73. P. Clements, D. Garlan, R. Little, R. Nord, and J. Stafford, in *25th International Conference on Software* (IEEE, 2003), p. 740.
74. https://github.com/GEECS-BELLA.
75. https://sites.google.com/a/lbl.gov/geecs/installation.
76. https://learn.ni.com.
77. https://www.ni.com/en-us/innovations/white-papers/18/introduction-to-the-distributed-control-and-automation-framework.html.
78. https://epics-controls.org/epics-users/projects.
79. https://epics-controls.org/resources-and-support/documents/epics-faq.
80. https://areadetector.github.io/areaDetector.
81. G. White, M. Shankar, A. N. Johnson, M. L. Rivers, G. Shen, S. Veseli, K. Shroff, M. Sekoranja, T. Cobb, T. Korhonen, D. G. Hickin, H. Junkes, M. G. Konrad, R. Lange, S. M. Hartman, K.-U. Kasemir, M. Pearson, K. Vodopivec, L. B. Dalesio, M. Davidsaver, D. Zimoch, and M. R. Kraimer, in *10th International Particle Accelerator Conference (IPAC2019)* (JACoW Publishing, 2019), p. 1216.
82. https://www.clf.stfc.ac.uk/Pages/EPAC-introduction-page.aspx.
83. https://epics-controls.org/support.
84. https://controlssoftware.sns.ornl.gov/training/2019_USPAS/#prep.
85. https://sfeister.github.io/sidekick-epics-docs.

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press

86. https://tango-controls.readthedocs.io/en/latest.
87. https://www.tango-controls.org/about-us/#History.
88. https://www.tango-controls.org/community/forum.
89. https://tango-controls.readthedocs.io/en/latest/reference/glossary.html.
90. A. Pope, *The CORBA Reference Guide: Understanding the Common Object Request Broker Architecture* (Addison-Wesley Longman Publishing, 1998).
91. https://www.omg.org/spec/IDL/4.2.
92. https://tango-controls.readthedocs.io/en/latest/overview/overview.html#what-is-tango-controls.
93. https://tango-controls.readthedocs.io/en/latest/tools-and-extensions/built-in/pogo/index.html.
94. P. V. Mockapetris, *Domain Names - Implementation and Specification* (1987).
95. https://gitlab.esrf.fr/limagroup/lima.
96. https://www.tango-controls.org/partners/institutions.
97. J.-L. Paillard, J.-L. Veray, M. Pina, and J. Froment, in *32nd Tango Collaboration Meeting* (2018).
98. L. Schrettner, in *32nd Tango Collaboration Meeting* (2018).
99. N. Weiße, L. Doyle, J. Gebhard, F. Balling, F. Schweiger, F. Haberstroh, L. D. Geulig, J. Lin, F. Irshad, J. Esslinger, S. Gerlach, M. Gilljohann, V. Vaidyanathan, D. Siebert, A. Münzer, G. Schilling, J. Schreiber, P. G. Thirolf, S. Karsch, and A. Döpp, High Power Laser Sci. Eng. **11**, e44 (2023).
100. K. Dupraz, M. Alkadi, M. Alves, L. Amoudry, D. Auguste, J.-L. Babigeon, M. Baltazar, A. Benoit, J. Bonis, J. Bonenfant, C. Bruni, K. Cassou, J.-N. Cayla, T. Chabaud, I. Chaikovska, S. Chance, V. Chaumat, R. Chiche, A. Cobessi, P. Cornebise, O. Dalifard, N. Delerue, R. Dorkel, D. Douillet, J.-P. Dugal, N. El Kamchi, M. El Khaldi, E. Ergenlik, P. Favier, M. Fernandez, A. Gamelin, J.-F. Garaut, L. Garolfi, P. Gauron, F. Gauthier, A. Gonnin, D. Grasset, E. Guerard, H. Guler, J. Haissinski, E. Herry, G. Iaquaniello, M. Jacque, E. Jules, V. Kubytskyi, M. Langlet, T. Le Barillec, J.-F. Ledu, D. Leguidec, B. Leluan, P. Lepercq, F. Letellier-Cohen, R. Marie, J.-C. Marrucho, A. Martens, C. Mageur, G. Mercadier, B. Mercier, E. Mistretta, H. Monard, A. Moutardier, O. Neveu, D. Nutarelli, M. Omeich, Y. Peinaud, Y. Petrilli, M. Pichet, E. Plaige, C. Prevost, P. Rudnicky, V. Soskov, M. Taurigna-Quere, S. Trochet, C. Vallerand, O. Vitez, F. Wicek, S. Wurth, F. Zomer, P. Alexandre, R. Ben El Fekih, P. Berteaud, F. Bouvet, R. Cuoq, A. Diaz, Y. Dietrich, M. Diop, D. Pedeau, E. Dupuy, F. Marteau, F. Bouvet, A. Gamelin, D. Helder, N. Hubert, J. Veteran, M. Labat, A. Lestrade, A. Letresor, R. Lopes, A. Loulergue, M. Louvet, M. Louvet, P. Marchand, M. El Ajjouri, D. Muller, A. Nadji, L. Nadolski, R. Nagaoka, S. Petit, J.-P. Pollina, F. Ribeiro, M. Ros, J. Salvia, S. Bobault, M. Sebdaoui, R. Sreedharan, Y. Bouanai, J.-L. Hazemann, J.-L. Hodeau, E. Roy, P. Jeantet, J. Lacipiere, P. Robert, J.-M. Horodynski, H. Bzyl, C. Chapelle, M. Biagini, P. Walter, A. Bravin, W. Del Net, E. Lahera, O. Proux, H. Elleaume, and E. Cormier, Phys. Open **5**, 100051 (2020).
101. https://gitlab.com/tango-controls.
102. https://gitlab.in2p3.fr/noliprog/instrument.
103. R. Bourtembourg, G. Cuní, M. Di Carlo, G. Fatkin, S. James, L. Pivetta, J.-L. Pons, S. Rubio-Manrique, C. Scafuri, G. Scalamera, A. Senchenko, V. Sitnov, G. Strangolino, P. Verdier, and L. Zambon, in *17th International Conference on Accelerator and Large Experimental Physics Control Systems* (JACoW Publishing, 2020), p. 1116.
104. https://www.ni.com/en-gb/support/model.pcie-6320.html.
105. https://www.ni.com/en-us/shop/pxi/ni-pxi-modular-instrument-design-advantages.html.
106. http://wiki.zeromq.org/results:10gbe-tests-v432.
107. https://tango-controls.readthedocs.io/en/latest/tools-and-extensions/built-in/astor/index.html.
108. https://tango-controls.readthedocs.io/en/latest/tools-and-extensions/built-in/jive/index.html.
109. https://tango-controls.readthedocs.io/en/latest/tools-and-extensions/built-in/atkpanel/atkpanel.html.
110. https://www.taurus-scada.org.
111. https://webjive.readthedocs.io/en/latest/what_is_it.html.
112. https://waltz-docs.readthedocs.io/en/latest.
113. https://docs.adacore.com/aws-docs/aws.
114. https://plotly.com/python.
115. https://tango-controls.readthedocs.io/en/latest/development/advanced/rest-api.html.
116. R. Trebino, K. W. DeLong, D. N. Fittinghoff, J. N. Sweetser, M. A. Krumbügel, B. A. Richman, and D. J. Kane, Rev. Sci. Instrum. **68**, 3277 (1997).
117. https://lists.tango-controls.org/wws/arc/info.
118. https://www.tango-controls.org/community/slack.
119. https://tango-controls.readthedocs.io/en/latest/installation/index.html.
120. https://indico.physik.uni-muenchen.de/event/135.
121. M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. G. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. C. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons, Sci. Data **3**, 160018 (2016).
122. https://www.go-fair.org.
123. https://www.w3.org/TR/owl2-overview.
124. M. Molinaro, M. Allen, F. Bonnarel, F. Genova, M. Demleitner, K. Graf, D. Morris, E. Solano, and A. Schaaff, arXiv:2111.14468 (2021).
125. M. G. Allen, S. Amodeo, F. Bonnarel, M. Molinaro, F. Genova, M. Romaniello, H. Heinl, A. Schaaff, and S. Lesteven, Bull. AAS **54**, 2022n2i028 (2022).
126. A. Weeks, G. Szabó, R. Hvězda, F. Gliksohn, and T. Ivănoaica, https://doi.org/10.5281/zenodo.6515903 (2022).
127. https://www.fairsfair.eu/implementation-adoption-stories.
128. https://www.forbes.com/sites/insights-cloudera/2022/09/30/the-case-for-fair-data/?sh=2535ea003328.
129. https://www.sheffield.ac.uk/openresearch/faircasestudies.
130. https://github.com/openPMD.
131. https://github.com/openPMD/openPMD-projects.
132. https://picmi-standard.github.io
133. https://github.com/openPMD/openPMD-CCD.
134. M. Könnecke, F. A. Akeroyd, H. J. Bernstein, A. S. Brewster, S. I. Campbell, B. Clausen, S. Cottrell, J. U. Hoffmann, P. R. Jemian, D. Männicke, R. Osborn, P. F. Peterson, T. Richter, J. Suzuki, B. Watts, E. Wintersberger, and J. Wuttke, J. Appl. Crystallogr. **48**, 301 (2015).
135. M. J. V. Streeter, C. Colgan, C. C. Cobo, C. Arran, E. E. Los, R. Watt, N. Bourgeois, L. Calvin, J. Carderelli, N. Cavanagh, S. J. D. Dann, R. Fitzgarrald, E. Gerstmayr, A. S. Joglekar, B. Kettle, P. McKenna, C. D. Murphy, Z. Najmudin, P. Parsons, Q. Qian, P. P. Rajeev, C. P. Ridgers, D. R. Symes, A. G. R. Thomas, G. Sarri, and S. P. D. Mangles, High Power Laser Sci. Eng. **11**, e9 (2023).
136. https://mlflow.org.
137. A. S. Joglekar and A. G. R. Thomas, J. Plasma Phys. **88**, 905880608 (2022).

https://doi.org/10.1017/hpl.2023.49 Published online by Cambridge University Press