The Journal of Open Source Software

LEAF: Laboratory Equipment Adapter Framework

Matthew Crowther[1], **Brett Metcalfe**[2,3,4], **Carlos Suarez**[5], **David Camilo Corrales**[6], **Jairo Alexander Astudillo Lagos**[8], **Anil Wipat**[1], **Timothy J. Rudge**[1], **Katherine James**[1], and **Jasper J. Koehorst**[2,7¶]

1 Newcastle University (NCL), Newcastle Upon Tyne, United Kingdom **2** Laboratory of Systems and Synthetic Biology, Wageningen University & Research, Wageningen, The Netherlands **3** Department of Earth Sciences, Faculty of Science, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands **4** Amsterdam University College, Amsterdam, The Netherlands **5** Departamento de Telemática, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca **6** TBI, Université de Toulouse, CNRS, INRAE, INSA, Toulouse, France **7** UNLOCK, Wageningen University & Research and Delft University of Technology, The Netherlands **8** INRAE, UMS 1337 TWB, 135 Avenue de Rangueil, 31077 Toulouse, France ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 19 January 2026
Published: unpublished

Summary

As our understanding of complex (biological) systems within the (life) sciences increases, so does a continuous demand for data to maintain that understanding. New approaches for knowledge generation, real-time prediction and decision-making, such as machine learning, do not reduce this data deficit, instead requiring diverse and continuously updated data streams. Furthermore, to be reusable, machine-readable, and computationally tractable, such diverse datasets must be seamlessly integrated and efficiently processed across different sources, necessitating a form of standardization. Yet the amount and diversity of data required far exceeds what traditional manual data capture and curation approaches can provide, making them increasingly infeasible. Because of this, there is a clear requirement to automate or at least to digitize laboratory experiments further and then extract and formalize the resultant data. While laboratory automation and automated data extraction are not new ideas, the issue of vendor-specific software, processes, and data structures makes generic inter-equipment data extraction challenging because a "one-size-fits-all" approach is simply impossible with such diverse and disparate systems. Here, we propose the Laboratory Equipment Adapter Framework (LEAF), a modular open-source system that can be adapted to monitor different laboratory equipment and extract experimental data automatically. LEAF is flexible and scalable, enabling integration with various types of laboratory instruments. By leveraging reusable modules, the framework allows for rapid development of new adapters, minimizing the effort required to support additional equipment. Once deployed, LEAF operates with minimal human intervention, facilitating continuous and automated data collection. Data is standardized through automated data collection, ensuring interoperability across different systems and enabling more efficient data analysis and downstream processing. Thereby, supporting compliance with data standards at the instrument level, generating better metadata, and allowing the integration of data from diverse sources. Ultimately, LEAF addresses the challenge of fragmented laboratory automation by providing a unified, adaptable solution for extracting and structuring experimental data.

Introduction

The Third Industrial Revolution, characterized by the advancement of digital technologies, has transformed industries by integrating digital systems and automation (Maynard, 2015). Just as in the wider world, innovation in digitalization has propagated into the day-to-day running of most research laboratories (Rihm et al., 2024). Most laboratories now benefit from equipment that is capable of some form of autonomous data generation (e.g., autosamplers), significantly improving the efficiency of data-driven research (Holland & Davies, 2020). However, despite this achievement, much of the resulting information and data remain locked within proprietary systems, creating 'data silos' that hinder interoperability and integration (Denton et al., 2021). This generates a paradox: laboratory automation is needed to accelerate scientific discovery (e.g., Naugler & Church, 2019; Wenzel, 2023), but a fragmented ecosystem of proprietary software, protocols, and data structures hinders widespread use and adoption. Further fragmentation occurs when different vendors - where vendors also include in-house IT or R&D teams - design (bespoke) equipment with unique architectures, incompatible formats, and isolated workflows, all making data extraction and integration a persistent challenge. Where existing solutions exist, they often rely on costly vendor-specific middleware, further fragmenting the data landscape and hindering cross-platform compatibility.

Unlike traditional computing, which experienced a transformative shift by adopting standardized protocols (such as USB, TCP/IP, and common file formats), laboratory automation remains an imperfect landscape where interoperability is often an afterthought. In the computing world, standardization enabled the rapid exchange of information, fuelling the digital revolution. In contrast, the lack of common frameworks in laboratory automation forces researchers to rely on vendor-specific solutions or custom-built interfaces, leading to inefficiencies, high integration costs, and significant barriers to large-scale data-driven research (e.g., Kleerebezem et al., 2021). Although the open science movement has gained traction (McKiernan et al., 2016), many laboratory instruments still rely on manufacturer-specific closed-source software to access and process the data. This reliance often results in a black-box approach to data generation and processing, outputs in a manufacturer-specific terminology, and the possibility that systems become outdated at a software level due to costly upgrade methods. Other instruments still rely on outdated methods, such as manually transferring output files, often in manufacturer-specific formats (colloquially referred to as the 'Sneakernet') (Jaya et al., 2020). Likewise, proprietary formats and closed communication protocols continue to hinder interoperability and automation, posing significant challenges to further integration with modern automation and data-sharing frameworks.

If left unresolved, these issues will create bottlenecks for FAIRification - the process of adapting datasets to the Findable, Accessible, Interoperable and Reusable (FAIR) principles (Wilkinson et al., 2016) - and open science (Ramachandran et al., 2021; Woelfle et al., 2011). They will also hinder progress towards Industry 4.0 technologies (Morisson & Pattinson, 2019) that rely on (near) real-time data transmission, such as machine learning (ML) and/or artificial intelligence (AI) (Acosta-Pavas et al., 2024; Metcalfe et al., 2025) and digital twins (Bauer et al., 2021; Grieves, 2014; Knibbe et al., 2022), the latter being digital replicas of physical systems. But more importantly, it necessitates the experimentalist to be overly concerned (Stirling, 2024), rather than unconcerned, with equipment-specific details and computational methods. To be agnostic of where high-quality data comes from is indeed an attractive thought.

LEAF Architecture and Design

LEAF Introduction

To address these challenges, here we present the Laboratory Equipment Adapter Framework (LEAF), an open source, modular solution designed to bridge the gap between diverse laboratory devices and virtual, client-side storage, monitoring and analysis tools (Figure 1). In this context, the term adapter refers to the conversion of attributes (e.g., data, metadata, commands, etc.) from one format to another format compatible with Internet of Things (IoT) architectures. Written in Python 3.12, LEAF provides a flexible, adapter-based architecture that enables the seamless monitoring of laboratory equipment, allowing real-time data exchange, automated processing, and integration with Industry 4.0 frameworks. LEAF achieves this through a modular design, where an equipment-specific module (referred to as an EquipmentAdapter) acts as an interface composed of a "Controller" and "Interpreter". These modules facilitate the retrieval, transformation and transmission of event data (Figure 2), reducing the complexity surrounding the integration of laboratory devices and lowering the technical barrier for researchers and engineers to acquire data from laboratory systems. It is in these adapters that the instrument and run (meta)data, as well domain-specific detail can be captured and if a user so wishes for terms to be standardised depending on how the adapter has been developed. However, it is important to note that LEAF is not a replacement for the (proprietary) software provided by the manufacturer of a piece of laboratory equipment; instead LEAF acts as a common interface by which data and it's associated metadata can be extracted from a device. LEAF is not a replacement for the (proprietary) software provided by the manufacturer of a piece of laboratory equipment.

The overall goal of LEAF is to act as an intermediary between isolated laboratory equipment and client-side tools. LEAF should be laboratory equipment agnostic, providing a gateway by which data and metadata can be transmitted to client-side tools such as databases, machine learning, dynamic models, and other monitoring and analysis tools. The LEAF system transforms the disparate data from equipment into semi-standardized formats ready for consumption in the larger infrastructure.

Figure 1: The overall goal of LEAF is to act as an intermediary between isolated laboratory equipment and client-side tools. LEAF should be laboratory equipment agnostic, providing a gateway by which data and metadata can be transmitted to client-side tools such as databases, machine learning, dynamic models, and other monitoring and analysis tools. The LEAF system transforms the disparate data from equipment into semi-standardized formats ready for consumption in the larger infrastructure.

At the core of LEAF, there have been five key principles:

- **Open** - To increase transparency, accessibility and allow for community collaboration without vendor lock-in.
- **Interoperable** – To encourage and to facilitate the system's use in as many pieces of laboratory equipment as possible, the system should be interoperable. For instance, it should be operating system (OS) agnostic.
- **Modular** - The system should be composed of elements that perform a distinct function, reducing complexity when developing, debugging, and using.
- **Reusable** - Developed elements should be reusable for developing new elements to reduce effort and maximize returns through iterative improvements.
- **Persistent** - Once running, the system should persist until intentionally stopped. This persistence should include even when the equipment is switched off, no experiments are running, and errors occur (experimentally or within the LEAF system).

How adapters work

An adapter within LEAF follows a specific workflow. Each subsystem or module within an adapter performs a specific function, with its inputs and outputs known beforehand. Broadly, this follows the input, mapping, optional transformation, and output steps (Figure 2). The overall goal is to detect an event or state change on the equipment, such as an experiment starting or a measurement being taken, extract it, and then output this along with the required data to the larger infrastructure (outside of the adapter). This infrastructure may be an Internet of Things (IoT) architecture containing a time series database or a dashboard with live monitoring and data processing.

An overview of the LEAF adapter framework. A single adapter for a particular piece of equipment (EquipmentAdapter, also referred to as a controller) can contain multiple processes (ProcessModule), each with multiple phases (PhaseModules), depending on the equipment type and requirements. When events (e.g., measurements) occur on the equipment, these are detected (InputModule) and route the data through an appropriate process (ProcessModule) or phase (PhaseModule). Depending on the event, data may be transformed before being transmitted externally (OutputModule).

Figure 2: An overview of the LEAF adapter framework. A single adapter for a particular piece of equipment (EquipmentAdapter, also referred to as a controller) can contain multiple processes (ProcessModule), each with multiple phases (PhaseModules), depending on the equipment type and requirements. When events (e.g., measurements) occur on the equipment, these are detected (InputModule) and route the data through an appropriate process (ProcessModule) or phase (PhaseModule). Depending on the event, data may be transformed before being transmitted externally (OutputModule).

Reusability

A core goal of LEAF is reusability, achieved through two mechanisms: Modules and Core Adapters. Modules handle specific tasks, such as monitoring databases or outputting data via MQTT, while Core Adapters provide predefined structures and mappings for common equipment behaviours without being tied to specific devices, abstracting function to enable reuse.

Modules

LEAF was designed to simplify adapter development by providing reusable modules, reducing the need for developers to build adapters from scratch. Many types of equipment report events similarly, often writing measurements to files or databases or transmitting them via standard protocols such as Protobuf or OPC UA. LEAF prevents redundant implementation and accelerates development by offering pre-built modules that handle these common functions. Adapters are structured with functional slots, such as InputModules and OutputModules, allowing developers to search an existing repository for suitable modules. Once a match is found, they can integrate the module directly into the adapter, ensuring consistency and reducing effort.

Core Adapters

Unlike base adapters, which require developers to define how equipment events map to processes and outputs, Core Adapters leverage similarities across seemingly different equipment. For example, fed bioreactors and microplate readers run discrete experiments with defined "running" and "not running" states, unlike a Raman Spectrometer, which operates continuously (Figure 3). This allows for intermediate Core Adapters—such as a DiscreteExperimentAdapter—that can be reused across equipment with experiment states. LEAF enables modular function reuse and logical consistency by structuring adapters around control concepts rather than specific devices, leaving developers to focus only on equipment-specific data transformation.

Mapping Events to Processes

Laboratory equipment are not simply machines that continuously perform actions and take single measurements. Most equipment have discrete experiment windows (the machine starts and stops), can provide experimental metadata, identify when errors have occurred, perform multiple actions, and measure many factors. Capturing the metadata around experiments, including the control aspects, provides a better context for the experiment. With a better grasp of what occurred during an experiment, experimental and methodological protocols can include details that allow for increased replicability. Therefore, because the LEAF system is designed to extract data from many pieces of equipment, and each piece of equipment may perform multiple actions and be in various states, there needs to be a mechanism to route different events into appropriate workflows. For example, a measurement should not be processed in the same way as an error message.

When an input event is received from the equipment, such as a measurement or an error occurring, the adapter must determine how to handle it. This is done by mapping the input to a specific internal process (Figure 2) with the knowledge that the given data type represents a broad event on the equipment (*e.g.*, an experiment run). Within each process, one or more "phases" exist, representing a discrete step or action, such as initialization, measurement, or shutdown. These phases contain the logic for interpreting and transforming the event data. By breaking down equipment behaviour into modular components and mapping events to the correct context, the adapter can cleanly separate concerns, accurately reflect the equipment's state, and make sense of the asynchronous, varied events produced by complex laboratory systems—ensuring downstream systems receive timely, well-categorized, and actionable data. Furthermore, because some processes are common (most equipment have runs that can be considered 'discrete' as opposed to 'continuous'), this modularity enables reuse across adapters, *i.e.*, once a process is defined, it can be reused in similar functioning adapters.

Results

Installing LEAF

It is advisable before installing LEAF to create a virtual environment to manage and to reduce conflict between dependencies, please see https://docs.python.org/3/library/venv.html for the latest (or version dependent) documentation for creating a virtual environment in Python. LEAF can be installed via Python Package Index (PyPI), see https://pypi.org/project/leaf-framework/, or via cloning the git repository.

The user can select the adapter that suits their equipment, or develop their own adapter. Once LEAF and the adapters have been installed on the relevant device a user must give LEAF a text based YAML format config file that allows for laboratory or user-specific settings to be applied. For example, a user can define the output as either MQTT, a database (KeyDB), or as a file. Users can also define an alternative, or as a fail-safe system in the event of network outage or MQTT disconnect, the output of LEAF can be stored as file or as a database (KeyDB). If used as a means of fail-safe these alternative outputs store the data until such time as transmission via MQTT can resume. Other options can also be specified including the device metadata (device id, organistaion, institute, etc.) and requirements (time interval, file path, port, host, api tokens, etc.).

Using LEAF

LEAF can be started in one of two modes:

- **Configuration File Mode:** The application is launched directly with a configuration file specifying the adapters and output settings required to communicate with the laboratory equipment.
- **Web Interface Mode:** If necessary, a web interface is available for startup, providing an alternative method for configuration.

When using the configuration file mode, the application establishes a connection to the laboratory equipment during startup, and the resulting data stream is forwarded to the MQTT server for further processing. In the web interface, a configuration file can be uploaded via the configuration section, which triggers the startup of the appropriate adapters. Additionally, the web interface allows users to discover and install adapters directly from the adapter section on demand. During an adapter's run, it can be stopped, restarted, or the configuration file modified. This can be achieved by modifying the configuration file directly and restarting LEAF or through the web interface and adjusting the configuration file accordingly.

Several adapters have already been developed GitLab: leaf-adapters that allows for connections to be established with wide range of laboratory equipment, including the BioLector XT Microbioreactor from Beckman, MinKnow from Oxford Nanopore Technologies, the MAQ weather observatory in the Netherlands, and iRODS (Integrated Rule-Oriented Data System) for storage monitoring. These examples demonstrate that a large range of cross scientific domain equipment and systems can use LEAF, provided that (i) the Python programming language can be installed on a connected work station; (ii) information can be extracted via a programmatic approach (*e.g.*, via API, reading files, *etc.*); and (iii) that a network connection exists for the transmission of information. To ease their use each adapter provides an example configuration as well as a device metadata file that contains all the information needed to establish a connection. For example, some adapters require the user to provide an IP address while other adapters might require the user to define an access token or a time interval for data extraction. Due to the vast heterogeneity of laboratory equipment, new adapters can be developed as and when they are needed to support various devices and/or specific protocols. Once installed, these adapters become visible to the LEAF system and can also be shared through the community LEAF Marketplace, extending the reach of your work in developing such adapters.

Example abstract hierarchy the LEAF system may create. All adapters are based upon the base adapter (EquipmentAdapter) and provide structure. The core adapter contains more specific adapters, which are designed to be used by functional adapters and contain common equipment processes. Functional adapters use the core or the base adapter and are specific to the equipment.

Figure 3: Example abstract hierarchy the LEAF system may create. All adapters are based upon the base adapter (EquipmentAdapter) and provide structure. The core adapter contains more specific adapters, which are designed to be used by functional adapters and contain common equipment processes. Functional adapters use the core or the base adapter and are specific to the equipment.

Discussion

Issues relying on vendors for standardization

Experimental and observational methodologies are increasingly becoming sources of high velocity and high volume data. High throughput analytics, for instance, is responsible for tens to hundreds to thousands of measurements per day. If the manual extraction, processing, and curation of such datasets for a single device is impractical, then the same task for laboratories with multiple, diverse devices becomes impossible, necessitating some form of automation. Reducing user intervention can also benefit from negating potential shortages of skilled workforces. Yet, automation is not a cheap solution, especially for a single laboratory where integrating different devices and vendor specifics into a harmonious whole can be a large software project. Standardizing laboratory automation even at a low level—particularly at the hardware and middleware layers— requires significant vendor cooperation (Bonvoisin et al., 2020). Without direct vendor support, standardization efforts face substantial roadblocks. In an ideal world, laboratory equipment vendors would subscribe to agreed-upon standard protocols, data formats, data structures, and output formats and strategies allowing laboratory managers to integrate new devices into existing laboratory data architecture quickly. However, currently, this is not the reality, making it a bottleneck to achieving an interconnected laboratory landscape needed for the next generation of analysis. Unlike higher-level software solutions, low-level standardization cannot be easily achieved through abstraction alone, as it depends on access to proprietary communication protocols, firmware source code, and hardware interfaces.

Whilst some manufacturers are agreeable to the potential of open source, many understandably retain tight control over their ecosystems, restricting third-party access to device firmware or proprietary middleware. In some circumstances, the incentives to adopt standardized interfaces are lacking, especially as doing so could reduce a vendor's control over software licensing, data formats, and service agreements. All of which are potentially lucrative sources of revenue. Yet without open access, developing universal, vendor-agnostic integrations becomes nearly impossible, as even well-designed frameworks struggle to bridge compatibility gaps without explicit vendor cooperation. From this frustration, LEAF was conceived as a solution to a problem within an imperfect world, *i.e.*, the struggle to extract all the data needed to perform complex real-time analysis of large experimental data.

Bridging the gap with LEAF

Here, we have presented LEAF, an abstraction layer that unifies laboratory equipment, providing a real-time connection for subsequent data analysis, processing, and storage. By automatically capturing data from heterogeneous devices, LEAF overcomes challenges posed by vendor-specific software and manual data handling, streamlining the resultant research workflows. The key benefits of such a system include equipment and laboratory monitoring, experimental monitoring and optimization, and enhanced reproducibility. In the following, we will expand upon these benefits.

Laboratory and Equipment Monitoring

For laboratory managers and technical staff, LEAF enhances laboratory oversight by unifying data from various devices, thereby collecting real-time performance metrics that can potentially allow early detection of likely issues before they escalate into costly failures. In addition, LEAF can be used to offer real-time visibility of available and active equipment and its experimental or measurement status. This enables efficient scheduling, identification of underutilized resources, and automatic logging of experiment timelines and parameters. By centralizing these data sources, LEAF supports a proactive, data-driven approach to equipment management.

Enhancing Reproducibility

For researchers, the real-time autonomous collection of data from laboratory devices and its subsequent transmission can help reduce errors associated with manual data extraction, such as file or data copying and conversion. By providing the data in a standardized format, simple but common errors, such as switching the month and date in timestamps, can be easily avoided.

Furthermore, scientific research often suffers from reproducibility issues due to missing contextual details, as biological systems are highly sensitive to even minor variations. LEAF mitigates this by enabling comprehensive metadata capture, including experimental parameters, hardware changes, and deviations from the intended (experimental) setup. Preserving these details ensures researchers can accurately reconstruct experiments while assessing unforeseen variables, increasing confidence in results and improving the reliability of replication efforts.

Experiment Monitoring and Optimisation

For researchers and laboratory staff, LEAF also facilitates real-time monitoring, allowing researchers to track ongoing experiments with greater granularity and adjust when deviations arise. This not only accelerates the iterative process and enhances experimental efficiency but also shifts decision-making from a 'post-experimental' phase to a more practical 'during-experiment' phase. For instance, researchers and technicians can terminate an experiment or measurement run whose dataset(s) may be of low quality or unreliable before exhausting time, money, and samples. Moreover, LEAF's continuous data streams enable the development of digital shadows and ultimately digital twins — virtual models that optimizes and control physical equipment in real-time through utilizing real-time measurements to update the virtual model's state parameters. By bridging the gap between simulation and laboratory operations, digital twins reduce manual trial-and-error, supporting adaptive and data-driven experimental design.

Cloud if you want to, local if you don't

Overall, LEAF establishes a platform that meets the need for continuous, integrated data flow. As laboratories increasingly adopt automation and computational analysis, LEAF can be a critical bridge between experimental workflows and modern digital infrastructure. By breaking down data silos and promoting interoperability, LEAF lays the foundation for more scalable, adaptive, and intelligent laboratory systems.

Future work

Community

While the LEAF system, as presented, is a fully functional approach for automatic data retrieval from various pieces of laboratory equipment, presently, there are only a small number of adapters. These adapters represent the equipment available to the authors, but of course, there are a multitude of laboratory devices and equipment that could conceivably be included in LEAF. In fact, if the LEAF system is to be successful, new adapters must be developed by the community—ideally through an open, collaborative model that encourages broad participation and contribution (Woelfle et al., 2011). It is partly for this reason that the LEAF system is modular, allowing individual users to openly develop individual adapters for pieces of equipment for the wider scientific and research community without having to re-develop the core functionality (*e.g.*, communication). Our aim is to foster a similar community spirit to the one that has developed around Home Assistant, or OpenHab, for IoT devices used in home automation (Kim et al., 2015), but for laboratory equipment. If a sufficient number of adapters are developed, especially for common as well as various types of equipment, the cost of laboratory automation will be reduced, as the wider community will bear it. Further future improvements are outlined in the following.

Utilities to deploy adapters

A critical functionality that we aim to develop rapidly is a mechanism by which existing adapters can be found. Our aim is for the user to be able to discover the repository hosting the adapter and install it via traditional development operations techniques (*e.g.*, GIT). Secondary tools are also being developed to alleviate manual processes. One tool is the "LEAF Marketplace", a centralized repository containing all existing adapters with advanced search features to allow users to easily find and import existing adapters. Another tool, the "Adapter Wizard," is a utility tool for creating configuration files (files needed to set up and run adapter instances) without requiring a user to define the documents manually.

Pan adapter management

LEAF requires that the technician or laboratory manager set up and maintain each device individually for larger laboratories with multiple versions of the same device. While maintenance should be minimal because adapters are designed to run indefinitely once set up, unavoidable issues such as power outages may make failure inevitable. In development is the LEAF Hub system (https://pypi.org/project/leaf-hub/), which is designed as an adapter management system that works locally within a laboratory, department, or organization. Users can manipulate adapters remotely with the LEAF hub, such as starting adapters or checking internal statuses. Upon completion, the LEAF Hub will simplify adapter management significantly, especially in laboratories with multiple adapters.

Closing the loop with control

Finally, the current version of LEAF is one-directional; data is transmitted from the device to a broker that can then be transmitted onwards to either back-end or front-end tools and services (*i.e.*, data processing, storage, *etc.*). The implementation of a data exchange adapter for the transmission of simulation results from models (*e.g.*, AI / ML, Digital Twins) to physical bioreactors, facilitating real-time corrective actions. This will require the design of standardized communication protocols, ensuring compatibility with control systems, and establishing a feedback loop to update simulations continuously. This approach can potentially improve process efficiency and robustness by enabling bioreactors to adjust parameters dynamically based on predictive simulations. In some instances and for some equipment, there is the potential to 'close the loop', allowing the transmission to send a prediction and the corrective action to be performed.

Conclusions

There is increasing demand for real-time data capture and generation to understand complex systems better, leading to scientific and laboratory equipment that benefits from automation. However, though they may be capable of generating data autonomously, these datasets may be effectively 'stuck' on such equipment, given limitations in interoperability and communication. To remove this impediment to real-time data integration here, the Laboratory Equipment Adapter Framework (LEAF) is presented. LEAF is a lightweight, open source, modular data acquisition solution designed to bridge the gap between diverse laboratory devices and virtual, client-side storage, monitoring and analysis tools (*e.g.*, time series databases, dashboards, analytical workflows, *etc.*).

Code availability

The Laboratory Equipment Adapter Framework, written in Python, is available from Gitlab at http://gitlab.com/labEquipmentAdapterFramework under an Apache License 2.0 license. It can be installed with the 'pip' package installer for Python from the command line in a terminal with the command `pip install leaf-framework` via the Python Package Index (PyPI), see https://pypi.org/project/leaf-framework/. Documentation about setting up, using, and developing new adapters is available at https://leaf.systemsbiology.nl. For those wishing to develop new adapters for laboratory equipment they can find assistance with the template available at https://gitlab.com/LabEquipmentAdapterFramework/leaf-adapters/leaf-template.

AI usage disclosure

Generative AI was used to convert the original latex version of the manuscript into markdown before submission.

Acknowledgements

Members of EU Horizon project 'BIOINDUSTRY 4.0: RI services to promote deep digitalization of Industrial Biotechnology - towards smart biomanufacturing' are thanked for their discussions and input.

Funding

The authors disclose receipt of the following financial support for the research, authorship, and publication of this article: European Union's Horizon 2020 research and innovation programme projects ‘RI services to promote deep digitalization of Industrial Biotechnology - towards smart biomanufacturing’ (BIOINDUSTRY 4.0, grant agreement n° 101094287). J.J.K acknowledges the Dutch Research Council (NWO), and Wageningen University & Research for their financial contribution to the UNLOCK initiative (NWO: 184.035.007). C.S. was funded through Call 823 - High-Level Training of Human Capital for the Cauca region. C.S. and D.C.C. acknowledge the support of the Ministry of Science, Technology, and Innovation (MinCiencias).

Author’s Contributions

Matthew Crowther: Software (Lead) - developed LEAF and example adapters (equal); Conceptualization; Writing - original draft and review and editing; Visualization (equal). **Brett Metcalfe**: Conceptualization; Writing - original draft and review and editing; Visualisation (equal). **Carlos Suarez**: Conceptualization. **David Camilo Corrales**: Conceptualization; Writing - review and editing. **Anil Wipat** Conceptualization; Writing - review and editing; Supervision. **Tim Rudge** Supervision. **Katherine James**: Supervision. **Jasper J. Koehorst**: Software - developed LEAF and example adapters (equal); Conceptualization; Writing - original draft and review and editing; Supervision.

References

Acosta-Pavas, J. C., Robles-Rodriguez, C. E., Griol, D., Daboussi, F., Aceves-Lara, C. A., & Corrales, D. C. (2024). Soft sensors based on interpretable learners for industrial-scale fed-batch fermentation: Learning from simulations. *Computers & Chemical Engineering*, *187*, 108736. https://doi.org/10.1016/j.compchemeng.2024.108736

Bauer, P., Stevens, B., & Hazeleger, W. (2021). A digital twin of Earth for the green transition. *Nature Climate Change*, *11*(2), 80–83. https://doi.org/10.1038/s41558-021-00986-y

Bonvoisin, J., Molloy, J., Häuer, M., & Wenzel, T. (2020). Standardisation of practices in open source hardware. *Journal of Open Hardware*, *4*(1). https://doi.org/10.5334/joh.22

Denton, N., Molloy, M., Charleston, S., Lipset, C., Hirsch, J., Mulberg, A. E., Howard, P., & Marsh, E. D. (2021). Data silos are undermining drug development and failing rare disease patients. *Orphanet Journal of Rare Diseases*, *16*(1), 161. https://doi.org/10.1186/s13023-021-01806-4

Grieves, M. (2014). *Digital twin: Manufacturing excellence through virtual factory replication* (p. 7) [White {Paper}]. Florida Institute of Technology. https://researchgate.net/publication/275211047

Holland, I., & Davies, J. A. (2020). Automation in the life science research laboratory. *Frontiers in Bioengineering and Biotechnology*, *8*, 571777. https://doi.org/10.3389/fbioe.2020.571777

Jaya, A. C., Safitri, C., & Mandala, R. (2020). Sneakernet: A technological overview and improvement. *2020 IEEE International Conference on Sustainable Engineering and Creative Computing (ICSECC)*, 287–291. https://doi.org/10.1109/ICSECC51444.2020.9557509

Kim, S.-M., Choi, H.-S., & Rhee, W.-S. (2015). IoT home gateway for auto-configuration and management of MQTT devices. *2015 IEEE Conference on Wireless Sensors (ICWiSe)*, 12–17. https://doi.org/10.1109/ICWISE.2015.7380346

Kleerebezem, R., Stouten, G., Koehorst, J., Langenhoff, A., Schaap, P., & Smidt, H. (2021). Experimental infrastructure requirements for quantitative research on microbial communities. *Current Opinion in Biotechnology*, *67*, 158–165. https://doi.org/10.1016/j.copbio.2021.01.017

Knibbe, W. J., Afman, L., Boersma, S., Bogaardt, M.-J., Evers, J., Van Evert, F., Van Der

Heide, J., Hoving, I., Van Mourik, S., De Ridder, D., & De Wit, A. (2022). Digital twins in the green life sciences. *NJAS: Impact in Agricultural and Life Sciences*, *94*(1), 249–279. https://doi.org/10.1080/27685241.2022.2150571

Maynard, A. D. (2015). Navigating the fourth industrial revolution. *Nature Nanotechnology*, *10*(12), 1005–1006. https://doi.org/10.1038/nnano.2015.286

McKiernan, E. C., Bourne, P. E., Brown, C. T., Buck, S., Kenall, A., Lin, J., McDougall, D., Nosek, B. A., Ram, K., Soderberg, C. K., Spies, J. R., Thaney, K., Updegrove, A., Woo, K. H., & Yarkoni, T. (2016). Point of view: How open science helps researchers succeed. *eLife*, *5*, e16800. https://doi.org/10.7554/eLife.16800

Metcalfe, B., Acosta-Pavas, J. C., Robles-Rodriguez, C. E., Georgakilas, G. K., Dalamagas, T., Aceves-Lara, C. A., Daboussi, F., Koehorst, J. J., & Corrales, D. C. (2025). Towards a machine learning operations (MLOps) soft sensor for real-time predictions in industrial-scale fed-batch fermentation. *Computers & Chemical Engineering*, *194*, 108991. https://doi.org/10.1016/j.compchemeng.2024.108991

Morisson, A., & Pattinson, M. (2019). *Industry 4.0. : A policy brief from the policy learning platform on research and innovation* (p. 22). Interreg Europe Policy Learning Platform.

Naugler, C., & Church, D. L. and. (2019). Automation and artificial intelligence in the clinical laboratory. *Critical Reviews in Clinical Laboratory Sciences*, *56*(2), 98–110. https://doi.org/10.1080/10408363.2018.1561640

Ramachandran, R., Bugbee, K., & Murphy, K. (2021). From open data to open science. *Earth and Space Science*, *8*(5), e2020EA001562. https://doi.org/10.1029/2020EA001562

Rihm, S. D., Bai, J., Kondinski, A., Mosbach, S., Akroyd, J., & Kraft, M. (2024). Transforming research laboratories with connected digital twins. *Nexus*, *1*(1), 100004. https://doi.org/https://doi.org/10.1016/j.ynexs.2024.100004

Stirling, J. (2024). Open instrumentation, like open data, is key to reproducible science. Yet, without incentives it won't thrive. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, *382*(2274), 20230215. https://doi.org/10.1098/rsta.2023.0215

Wenzel, T. (2023). Open hardware: From DIY trend to global transformation in access to laboratory equipment. *PLOS Biology*, *21*(1), e3001931. https://doi.org/10.1371/journal.pbio.3001931

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18

Woelfle, M., Olliaro, P., & Todd, M. H. (2011). Open science is a research accelerator. *Nature Chemistry*, *3*(10), 745–748. https://doi.org/10.1038/nchem.1149