doi:10.2533/chimia.2025.36 Chimia 79 (2025) 36–45

The openBIS Digital Platform for Instrumentation and Data Workflow in the Analytical Laboratory

Yousuf Hemani[a,b], Kilian Koch[a], Oscar Mendo Diaz[a,b], Anusch Bachhofner[a], Simone Baffelli[a], and Davide Bleiner[a,b*]

Abstract: The management of scientific data plays a key role in all research areas and has increased in importance. Providing researchers with customizable data management tools is crucial for effectively managing data according to the FAIR principles. These principles have been defined by Wilkinson *et al.* in 2016, which describe how scientific data should be managed.[1] To support the specific needs of researchers at Empa, openBIS[2] was chosen as a FAIR compliant data management platform. OpenBIS is an Electronic Laboratory Notebook (ELN) and Laboratory Information Management System (LIMS) developed at ETH. The commissioning of this platform for the case of an analytical chemistry lab presented multiple challenges. In this paper, solutions to adapt openBIS as a digital platform to integrate the laboratory data workflow in chemical analysis and for spectroscopy instruments are presented. Two laboratory projects as case studies are described, consisting of a data pipeline and a complex dashboard for data collection, visualization and interaction. These examples show a successful integration of the data management platform in accordance with the FAIR data guidelines along with maximizing efficiency for laboratory personnel.

Keywords: ELN · FAIR · LIMS · OpenBIS · Smart lab

Yousuf Hemani is a doctoral candidate working on high energy lasers in the Laboratory of Advanced Analytical Technologies at the Swiss Federal Institute of Material Sciences and Technology (Empa).

Anusch Bachofner is a project manager and data scientist working in scientific IT at Empa. She led the project of the openBIS introduction in Empa labs and is supporting researchers in data management topics.

Kilian Koch worked as a digitalization specialist in the Laboratory of Advanced Analytical Technologies at the Swiss Federal Institute of Material Sciences and Technology (Empa).

Simone Baffelli received his MSc degree in electrical engineering from ETH Zurich in 2013 and obtained a PhD from the same institution in 2018 with a dissertation on terrestrial radar interferometry. He was a postdoctoral researcher with the Swiss Federal Laboratories for Materials Science and Technology (Empa), where he worked on geostatistical analysis, sensor calibration, and data assimilation techniques for atmospheric monitoring networks. He currently works at Empa as a research software engineer.

Oscar Mendo Diaz completed his MSc degree at the University of Zurich, in collaboration with the Swiss Federal Laboratories for Materials Science and Technology (Empa), with his Master's thesis in the field of chlorinated paraffins. Afterwards, he was employed at Empa as a scientist and extended his stay to pursue a PhD in the same field. Since then, he has developed innovative methods for the extraction and evaluation of data concerning chlorinated paraffins and their transformation products.

PD Dr. habil. Davide Bleiner is the head of Advanced Analytical Technologies at Empa and Adjunct Lecturer at ETH and UZH. He has pushed an innovative digitalization plan in the analytical lab for FAIR and lab virtualization.

*Correspondence: Dr. D. Bleiner, E-mail: davide.bleiner@empa.ch
[a]Swiss Federal Laboratories for Materials Science & Technology (Empa), Überlandstrasse 129, CH-8600 Dübendorf, Switzerland; [b]University of Zurich, Winterthurerstrasse 190, CH-8056 Zurich, Switzerland.

1. Introduction

Many research fields, from materials science to environmental sciences or metrology,[3–6] are resorting to the implementation of robust methods for reliable acquisition and analysis of qualitative and quantitative data. Besides for archiving purposes, in bio/chemical, physical, material sciences and related disciplines, the accurate handling of instrumentation data is key for insights.[7–8] Within the field of material sciences, data is pulled from multiple experiments, and thus often involves heterogeneous datasets with a variety of important key parameters (metadata).[9] The information contained in the form of metadata is crucial for post-processing analysis and the organization of the data in the laboratory.[10]

Traditionally, one would acquire the relevant data with an instrument and store it on the lab computer, analyze it and record the supporting metadata and the findings in a paper notebook. This has been the way to do things in an academic laboratory since the beginning.[11] An epochal shift has been seen towards digitalization techniques and paperless notebooks to make the process more agile by the use of Electronic Lab Notebooks (ELNs) and Laboratory Information Management Systems (LIMSs).

Digitalization techniques facilitate the monitoring of the instrument performance and calibration, efficient and automated recording of data over longer periods of time, storage of data on different platforms, post processing analysis and most importantly efficient visualization of data. The findings can be digitally recorded along with the relevant metadata and the standard operating procedures (SOPs).

An ELN addresses the challenge of information and data management where large amounts of data are generated and stored.[12] It aids in data retrieval, reproducibility, data sharing and data tracking for intellectual property and patents. It also helps in defining methodologies to collect data and preserving metadata for measurements. Overall, this optimizes a laboratory's workflow and minimizes the time spent. An ELN becomes even more practical if combined with a LIMS.[13]

A LIMS is a digital platform for handling data, methods, and inventory for routine analysis in academic laboratories and industries.[14] Its power lies in the effortless integration of aspects such as sample management, stock management, data archiving and reporting from multiple users. For several years LIMSs have been used to streamline operations, increase time and process efficiency for data management and acquisition, and to improve access by stakeholders with regards to compliance regulations.[15]

The selection of an ELN/LIMS system depends on several factors. A primary aspect is the cost-to-benefit ratio.[16] Additionally, a good ELN/LIMS is easily customizable and adaptable to the laboratory needs with reliable data and inventory management capabilities.[17] Depending on the application, it should be able to integrate instruments and measurement processes where the direct acquisition, storage and visualization of data can take place and is easily accessible by web and hand held devices. This is useful for complete monitoring of the instrument and laboratory facilities, method validation as well as data measurements by several users with regards to defining the SOPs.[18] The end goal is production of high-quality useful data and its access and shareability towards the respective stakeholders.

Several examples of both ELNs[19] and LIMSs[20] exist, either commercially licensed or open source. Some examples are Leaf LIMS for synthetic biology,[21] appMAGI for clinical diagnostics,[22] HaIS[23] and SIGLa[24] for both small and large scale laboratory applications. Many institutions such as IUSM[25] and EPFL[26] have initiated efforts for institute-wide implementation of ELN/LIMS system.

In 2011, openBIS (open Bio-Informatic System) was released as an open-source information system (IS). It was originally developed for biological and life science laboratories by ETH Zurich.[27] openBIS was created to provide ELN/LIMS software with exceptional features and customizable abilities, which also facilitates adhering to the FAIR data management guidelines. The FAIR principles describe the requirements of scientific data for a sustainable and ethical approach.[28] The FAIR acronym refers to *Findable, Accessible, Interoperable* and *Reusable*. Management and stewardship of scientific data according to these guiding principles is impossible without a dedicated digital tool for laboratory information management.[29] Due to the highly configurable structure of openBIS, it can be adapted to most scientific fields. Since 2021, the researchers at Empa have been provided with continuous support to utilize the ELN/LIMS system openBIS for digitalization of their respective laboratories which also facilitates adherence to the FAIR principles.

The openBIS platform provides laboratory experiment and project management, protocol management, inventory management, data storage and analytics, and application programming interfaces.[30] Examples of the adoption of openBIS by different institutes for several unique projects can be found in the scientific literature.[31-34]

The initial challenge is to adapt the platform to a specific context. Secondly, it is crucial to train users on the platform and encourage users to regularly record their data. Two case studies are chosen to be digitalized using openBIS for the Laboratory of Advanced Analytical Technologies at Empa. The case studies are inspired by an ongoing development in academia[35] to achieve the so-called 'Smart lab' for various analytical chemistry activities in the department. A 'Smart Lab' is defined as a laboratory environment, which leverages advanced tools such as automation techniques and data management techniques for increasing the accuracy, efficiency, productivity, and safety in scientific research by the implementation of an intelligent laboratory workflow.

The first case study is about the adaptation of openBIS for the automatic data management of Mass Spectrometry data. The second case study is about implementing a live monitoring and diagnostic dashboard for the high-energy Laser facility. Similar implementations and case studies existing in scientific literature demonstrate how large amounts of unstructured data generated from various experiments can benefit from an automated structure, storage and access. This is achieved by using an open source ELN such as LabTrove;[36] which facilitates efficient recording and analysis, project coordination and collaboration, and aids in the publication process. In addition, monitoring and control of large-scale laser facilities is crucial for maintenance and continuous user experiments. Efforts are being made to create a long-lasting digital infrastructure for laser facilities where users can be provided with reliable and organized metadata and online analysis facilities. Automated measurement of experimental parameters at regular intervals and continuous data collected from a suite of implemented diagnostics also enable the development of feedback and control loops for the instrument, eliminating the repeated so-called 'human interaction'.[37]

The aim of this paper is to highlight the capabilities of openBIS for analytical chemistry and instrumentation, especially when combined with advanced frameworks using the Python programming language. A guideline to successful digitalization projects for laser facility management using openBIS in combination with other diagnostic elements to achieve modern solutions of quality control and visualization is provided.

The paper is organized as follows: In section 2, the tool openBIS is introduced and its digital hierarchical structure is described. The platform's features and a definition of guidelines to the laboratory data structure and workflow is given in sections 3 and 4. Two examples demonstrating the automation capabilities with openBIS are shown and discussed in-depth in section 5. Finally, the main insights are summarized in section 6.

2. openBIS Structure

The tool openBIS was developed with the intent of providing an open-source platform for data management in academic laboratories. Due to its configurable nature, this system can be modified to suit almost any field to provide a data structure which ensures FAIR compliance. However, this configurability also presents a hurdle that must first be overcome, as system administrators need to carefully choose how to set up the structure in openBIS.

By default, an openBIS instance uses a hierarchical structure as illustrated in Fig. 1. Starting from the topmost container called **space**, it contains one or more **projects**. A space is mostly used for access control in the case of multi-group openBIS settings, where certain users should only access some of the data and metadata.

A **project** in turn contains one or more **experiments**. An experiment can contain structured metadata in the forms of fields that are displayed in the ELN. These metadata help in searching and organizing experiments. Besides standard experiment types, openBIS is extensible and allows defining custom experiment types with user-defined metadata fields.

Attached to an experiment are **samples**. These are also configurable through custom sample types. Samples (or objects) are used to represent various experimental units, such as chemicals, physical samples or inventory items such as experimental instruments.

Besides being embedded in an experiment, samples can also be connected to each other through parent–child relationships in order to represent more complex graphs of relationships. The parent–child hierarchy is used to organize data in a structured way such that complex experimental workflows can be represented seamlessly.

Finally, both samples and experiments can be associated with one or more **datasets**, which contain measurement data or any other type of file-based data. The metadata stored in datasets are also configurable and here too openBIS already offers some predefined types.

This basic hierarchical structure cannot be changed in openBIS. In the specific case of our research laboratory, three research groups share one openBIS instance. Some materials and methods of well-known and often used objects are shared across groups, but each group also has specific compartmentalized spaces, such that the platform remains clean and tidy and data access stays within groups. Additionally, there is a common space which can be accessed by members of all groups where joint projects are held. Finally, openBIS allows fine-grained user access management down to a project basis with read or read/write permissions.

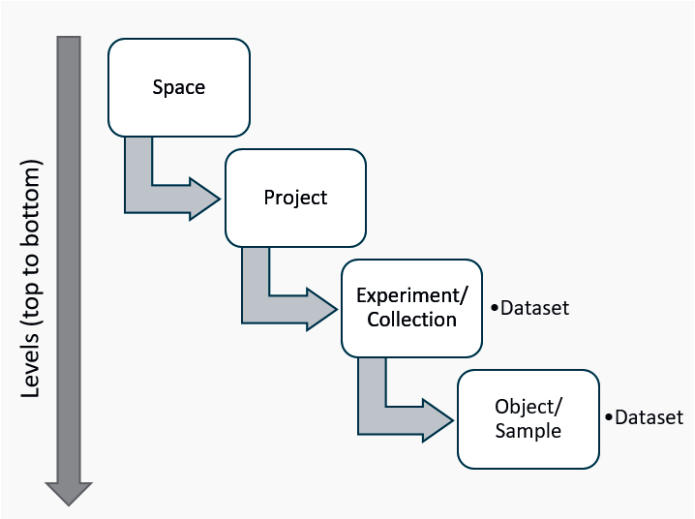

Fig. 1. Hierarchy structure of openBIS Instance.

The digital structure and hierarchy facilitate efficient data retrieval and its effective use.

3. openBIS Functionalities

The openBIS system comes with three important functionalities for digitalizing workflows; the extract-transform-load (ETL) function, the application programming interface (API) feature *e.g.* pybis package, and the electronic lab notebook (ELN) and laboratory information management system (LIMS), all of which are explained in this paper.

An extract-transform-load function or ETL is a crucial concept in data warehousing.[38] The first step of an ETL function in the context of electronic lab notebooks is to extract raw data from an external source such as an analytical instrument. In the second step the raw data undergoes transformation; fitting the needs defined by the researcher. The third step is data storage and archiving such that it is accessible and ready for analysis. For the implementation of the second step, administrators as well as users should be trained specifically on data management principles to provide clean and well-described data input. The openBIS ETL function serves as a pipeline for automatized data upload and metadata extraction. An openBIS ETL pipeline ('dropbox' in the openBIS jargon) is a Java or Jython script deployed on an openBIS server. The script has access to the internal openBIS API and can create arbitrary data and metadata on the ELN LIMS system. Once deployed, it monitors a folder for incoming data and if the data matches pre-defined criteria, it extracts metadata and derived data from the uploads and stores them in openBIS. In this way, metadata for experimental data can be extracted automatically from the provided measurements without any need of user interaction.

The openBIS framework allows any number of data objects to be created in openBIS *via* either the web based interface or by using various APIs (Python, JS, Matlab, REST). The objects in openBIS are used to represent various research items such as projects, experiments, samples, datasets, *etc.* One can define a structure or systematic workflow of a specific project or experimental setup *via* different kinds of folders containing metadata that are called 'projects', 'experiments' and 'experimental step'. One can also create a dedicated database of materials or methods *via* the so called 'Collection folder'. The so called 'Materials folder' can contain the information of samples commonly used for experiments and the so called 'Methods folder' can contain a list of instruments for performing experiments. The characteristics for all samples logged can be defined in openBIS *via* the so called 'Object type'. The object type is basically a blueprint for creating specific objects. They determine the template of the object properties either pre-defined or custom and ensures consistency in defining similar objects. For describing how to operate an instrument, a set of standard operating procedures/protocols (SOPs) can also

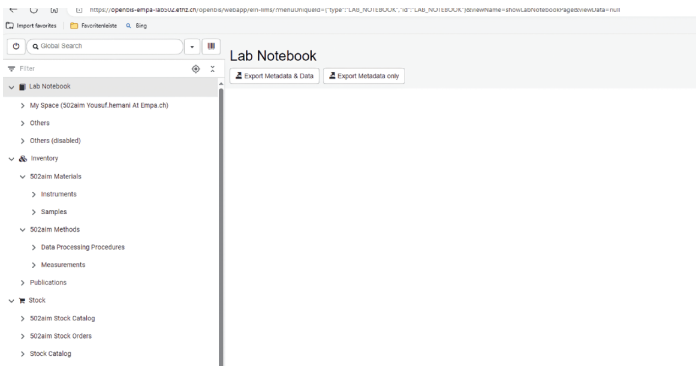

Fig. 2. The screenshot of the openBIS platform shows its UI and respective folders. More screenshots are available in the official documentation.[39]

be defined. A screenshot of the openBIS platform displaying the important folders is shown in Fig. 2.

The data generated by instruments and their respective metadata can also be uploaded to openBIS *via* the user interface (UI) or the implementation of a feature that is called 'dropbox function', as developed by ETH. The data saved can have the properties/parameters including units properly described with a standard file format. This can then be used for data analysis on openBIS *via* the implementation of a dashboard architecture and programming scripts using Python or MATLAB programming languages.

Although the general operating structure of openBIS comes with default admin settings and features, such as folder structures and standard objects and object types; the structure can and mostly needs to be modified before the instance is set up for a specific laboratory, depending on the localized needs.

Especially when working in cross-functional teams or multiple laboratories with varying analytical settings, but with a common base between all parties, one should collaborate on a shared openBIS instance to facilitate interoperability. This characteristic is further explained in-depth in this paper as it is tightly connected to the data structure.

The openBIS servers (Application server and Data Store Server) provide access to the ELN for users as well as an additional admin user interface, which is a separate front-end application for instance administrators. The ELN is the front end where most users will interact with the system. Administrators have another interface called 'Admin interface' to configure openBIS. Additionally, users and administrators can use a REST (JSON-RPC) API, which allows access to all openBIS functionalities by leveraging the use of various programming languages, *e.g.* Python commands.

Furthermore, by using the so-called 'dropbox function' administrators can configure ETL scripts to directly upload files from the file explorer to a specific openBIS object. The ETL scripts allow for more complex logical transactions including all openBIS functions. The official openBIS documentation[39] is a good resource to find more details about the functionality and features provided by the platform such as the ELN, the dropbox function, the admin API and additional tools. Also, to improve content sharing between different laboratory instances and improving interoperability, continuous efforts are being made in the ETH domain with round the clock technical support available to both the users and admins.

3.1 *Setting up a Lab Instance*

To mirror process steps or sample properties of a research project and laboratory experiments in openBIS, their information needs to be added in the form of an object (or data set). To better represent the metadata specific to various entities, like samples, chemicals, inventory items *etc*, it is often advisable to define custom objects, datasets and experiment types. These types are similar to classes or datatypes in programming languages and their metadata fields can or must be filled by the user for a certain object type. Each metadata field is assigned an internal name, a user-readable label and a data type, which can be selected from several basic types such as numbers, strings, dates or entries from a controlled vocabulary.

The latter presents a powerful way to structure laboratory metadata, as it unifies different notation standards by only allowing values from a predefined set.

Additionally, administrators can assign property validation and dynamic property scripts to object properties. Property validation scripts can be used to check if the data entered by the user conforms to a specific set of constraints defined by a Python script or a Java method. Dynamic properties can be used to compute property values from other object properties using a Python script.

Although openBIS comes with several pre-defined objects, most administrators will need to create additional object types according to their object needs (*e.g.* projects, processing protocols, chemicals, instruments). For the purpose of an analytical chemistry laboratory with custom-made instruments, additional object types are created which are represented in Tables 1 and 2.

Table 1. Example object types of the chemical laboratory.

Measurements	Chemicals	Instruments	Samples	Information
LC MS CP	External chemicals	Instrument	Sample external	Publication
GC/MS	Internal chemicals	Instrument component	Sample internal	Staff
		Instrument service	Sample preparation	Supplier
			Sample type	Target
				Protocol SOP

Table 2. Dataset objects of the spectroscopy laboratory.

Information	Quality	Diagnostics	Measurements
Publication data	Standard operating principle	Spectroscopic diagnostics	Mass spectrometry data
	Safety data sheet	Oscilloscope diagnostics	Laser spectroscopy data
	Certificate	Laser Diagnostics image	Oscilloscope data
			Processed data

Although this list is not exhaustive since new processes can be introduced, they do include the most useful laboratory objects to streamline a workflow. Furthermore, some objects were specifically introduced for a specific type of analysis, which turned out to be helpful to facilitate the object search process.

3.2 *openBIS Implementation*

In openBIS, the process of importing information that is used to define the workflow structure and data organization in the instance is called 'Master Data Import'. This includes setting up schemas for metadata, controlled vocabularies, experiment types, sample types, *etc*. To create these objects and object types in openBIS *via* master data import based on a defined schema, the user can utilize either the Application Programming Interface (API) or the web-based openBIS admin interface. The web interface is the most convenient due to its graphical representation. Before defining the structure of the objects, one needs to conceptualize the desired structure, workflow, and properties. Thus, it is recommended to create a draft of the workflow, the desired (or necessary) metadata and processing. A workflow draft is illustrated as an example in Fig. 3. The scheme shows a customizable workflow in an analytical laboratory. The laboratory user modifies the sample using two processes. Each process uses a chemical to modify the sample. Afterwards, the processed sample is transferred to a laboratory instrument which is connected to a computer. The user then measures a specific property of the sample and transfers the data to the openBIS system.

For each of the shaded objects shown in Fig. 3, a set of metadata can be recorded and reported. When considering the sample as an example object, one must consider all properties such as weight, volume, concentration, *etc.* for workflow reporting. An

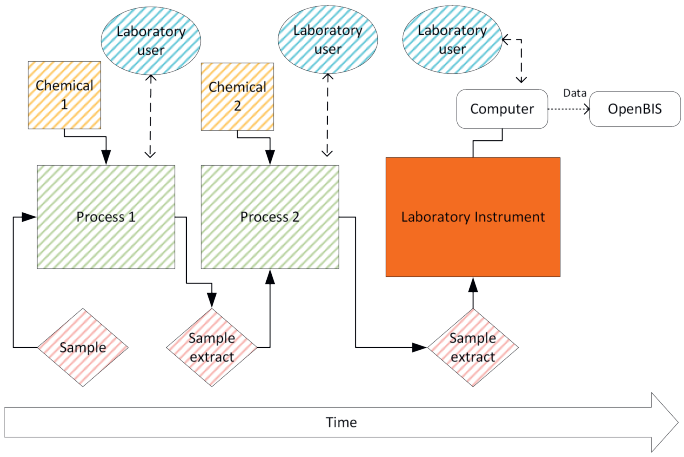

Fig. 3. Scheme of an example laboratory workflow of analytical chemistry.

Table 3. Example resources for a pesticide determination workflow.

Resource (see Fig. 1)	Example	Metadata
Process 1	Sample weighing	Units (*e.g.* gm)
Chemical 1	Diatomite	Symbol
Process 2	Soxhlet extraction	Temperature
Chemical 2	Ethanol	Concentration
Sample	Tobacco	Type
Laboratory Instrument	Liquid column mass spectrometer	Model Number

example set of explicit resources for an analytical workflow is shown in Table 3.

Processes can be defined with a selected granularity, *e.g.* a task such as filtration might be considered as not relevant to be reported on. Thus, it is recommended to rank sub-processes by relevance with their defined parameters for efficient data reporting and organization in openBIS.

4. Implementation of Data Workflow

The implementation of a typical data workflow for a laboratory experiment and its integration with openBIS is done in several steps as discussed in this section. Two specific application examples as case studies utilizing openBIS are described in the next section.

4.1 *Data Preprocessing*

As labs generally should and do keep track of their instruments, chemicals, reference materials *etc.* in structured formats (*e.g.* Excel, Access, Databases), it is helpful to write a parser and import script in a programming language supported by the openBIS API, which converts the user entries to openBIS –objects and stores them in the server. Another option is to use the XLS-batch register function of openBIS (view openBIS documentation for detailed information).[39] It allows users to download an empty template of the specific objects which can then be filled with properties and re-uploaded for creation of objects. Using a parser in combination with the API pybis package uploads the data efficiently in a single step. In contrast, the downside of using the XLS batch register is that data upload for creation of objects needs additional steps. Prior to parsing, most data needs cleaning. What is meant by data cleaning is the process of enforcing rigid notation standards on manually created ('messy') data.[40] This is the most sensitive and time-intensive step of data preprocessing, as it can very quickly lead to introducing incorrect data.

4.2 *Handling Missing Values*

When working with data originating from laboratory staff, it is common to expect human errors where incomplete, unidentifiable, or wrong data is entered or provided. It is crucial to think about how to handle these missing or flawed data values, as some missing properties will have more impact on the data integrity than others and can be problematic for data analysis. Ideally, the less critical information should be marked as optional from the beginning to ensure the recording of data is not hindered. Also, this can avoid creation of objects with insufficient data. For the properties which are considered necessary for the utility of data, they can and should be set as mandatory to record. This ensures all the key information is recorded. Additionally, where a key property is missing, assigning a sensible default value is crucial to maintaining the dataset's usability. For example, a placeholder value such as 'none', or '0' for text fields and numerical fields respectively, can be used. These data categorizing measures help in standardizing the dataset recording and reduces the risk of errors in dataset processing.

4.3 *Mapping Properties to Objects*

If objects are uploaded with the API (in this case using its python implementation, pybis), the mapping of a source property to the corresponding openBIS-object needs to be defined. This is mainly done using the pandas Python package[41] and the map function of data frame objects. Before mapping analytical properties with their respective openBIS property, they must be converted to the notation defined on openBIS. An example to demonstrate this concept is shown in Fig. 4.

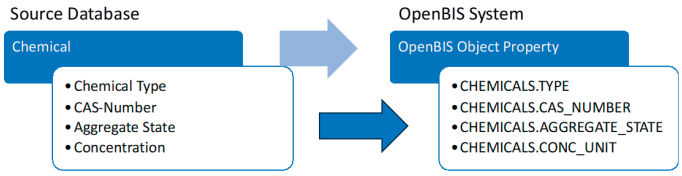

Fig. 4. Mapping of source data to openBIS data.

After conversion of the objects' original units to the previously created openBIS object property, we iterate through all data frame entries and create the objects in openBIS. For transparency, the openBIS properties are written with capital letters. It should be noted that ideally, the source data adheres to a standard such that it allows for data enrichment from an external database, such as PubChem.

4.4 *Datasets Upload with Respective Properties*

When migrating data to openBIS, it can be efficient to upload datasets in batches, especially when the datasets are easily identifiable (*e.g.* all *.csv-Files) and held in complex folder-structures. Using Python in combination with the API, we can iterate through all subfolders. If the name or content of the file holds metadata of the experiments in a structured readable format, it can be used as an additional input when creating the openBIS-object.

4.5 *Management and Retrieval of Information*

The web interface of openBIS allows users to easily manage their projects, experiments, samples, protocols, and objects. Especially when small changes need to be done to individual objects, it is usually faster *via* the web interface.

5. Application Workflows

In this section, two case studies of openBIS-implementations consisting of easy to complex workflows are highlighted. The first example will discuss the development of the Extract-Transform-Load (ETL) function for its use in efficient upload and storage of datasets to openBIS as generated from Mass Spectrometry experiments and the second example consists of a detailed guide through the architecture process of a user dashboard for live diagnostics and analytics of a high-power laser facility.

The following subsections give an overview of useful workflow blocks, such as creation of new objects, connection of objects and metadata extraction for specific formats. It should be noted that we consider spectroscopic data in a readable text-based format, but it can be applied to text-based files of any nature with a known structure.

5.1 *Case Study I: ETL for Analytical Data*

To address the local needs of our analytical chemistry lab, an ETL function (mentioned in section 2) for the automatic creation of openBIS objects based on Mass Spectrometry (MS) data was developed. The aim was to automate the upload of datasets from the MS instrument directly to openBIS instance, while extracting human-readable metadata for the openBIS user interface.

The files came in the default raw format, which is the file format used by a certain instrument manufacturer. The developed script was then tested using a local openBIS instance, which was deployed using the Docker image provided by the Scientific IT Services in ETH Zurich. A docker image is a full openBIS server in a single file, which can run as a portable executable file on any machine.

The standard workflow in openBIS involves creating objects with properties that identify and describe the experiments. Thus, the metadata should be available upon creation of this object. As many spectroscopic data formats are not readable with text editors but need special proprietary software or additional packages installed, it can be useful to structure filenames with information about the experiment itself. If this is not practical, a readable and structured text file can be additionally created which contains all metadata in a structured format, such as JSON, CSV or YAML. Sometimes, proprietary software allows the export of a machine-readable file, which holds experimental metadata. As the openBIS ETL plugins must use the Jython release of Python,[41] which is based on Python 2, it is unlikely that one can use publicly available packages to read specific metadata special formats.

Therefore, most of the information, rather than being extracted from the raw data, was encoded in the filename and then uploaded to openBIS.

For the case of the mass spectrometry data, the structured filename approach was used. The implementation was done according to the structure shown below, where PROJECT corresponds to the openBIS-internal immutable CODE-field of the ordinate project. Batch, sample type, injection and sample name were desired identification fields that in combination are unambiguously assignable to one measurement.

PROJECT_BATCH-SAMPLETYPE-INJECTION-SAMPLENAME.RAW

The developed Jython script splits the filename at underscores and parses the attributes sequentially, after checking for the validity of their values. The validity checks and transactions with openBIS API are shown in Fig. 5.

In openBIS, both metadata and data are managed effectively. When utilizing custom ETL processes, any upload results in an entry that is more readable and includes linked raw data. By hav-

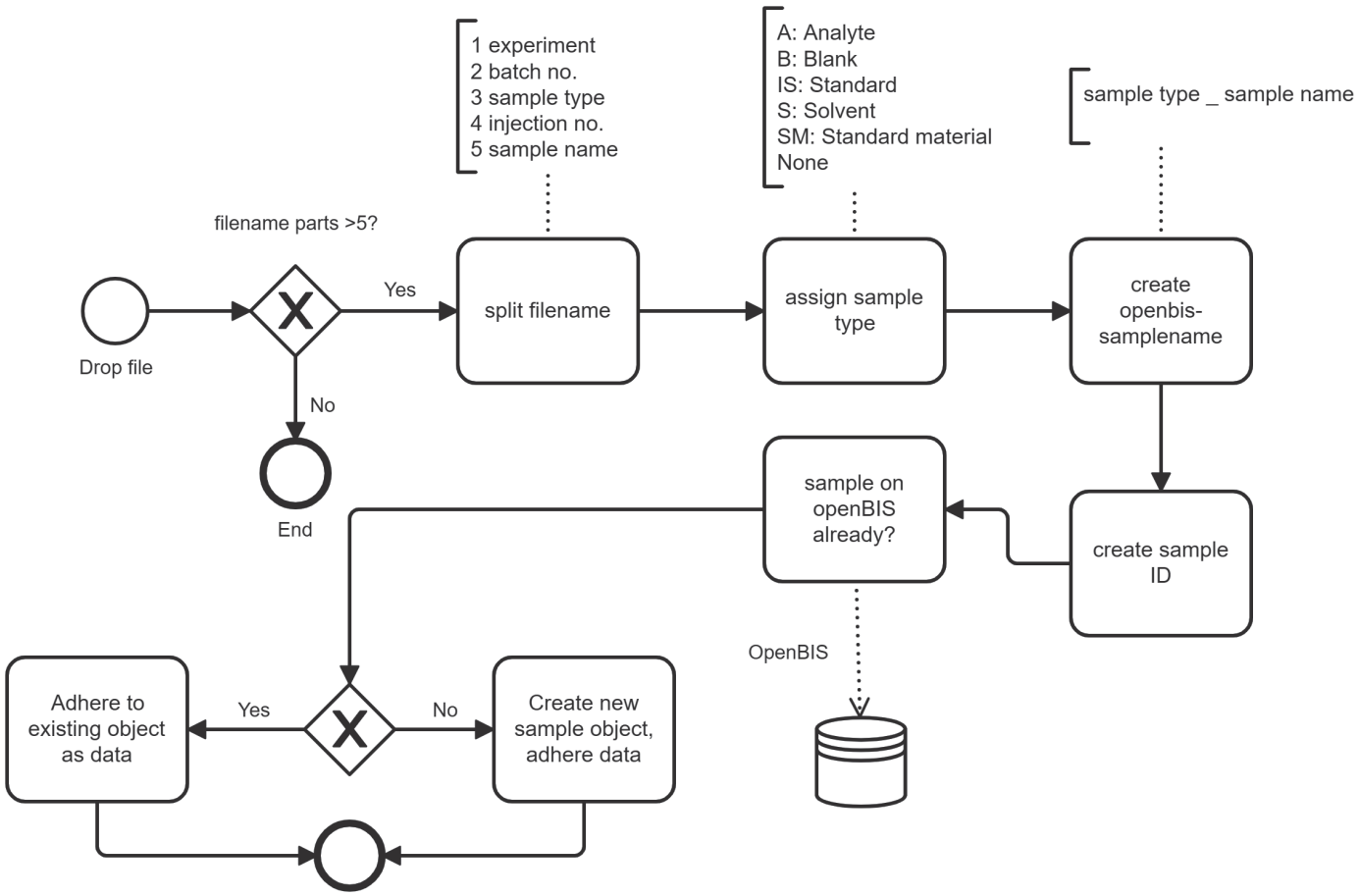

Fig. 5. Process scheme of openBIS-dropbox for MS-data

ing metadata and raw data interconnected, clarity and organization are significantly enhanced.

5.2 *Case Study II: Dashboard for Instrumentation*

The diagnostic elements installed in a laser facility such as cameras, spectrometers, oscilloscopes and energy meters, all push raw data, which gives information such as beam characteristics, *etc.* For the ongoing development of the terawatt laser facility,[43–45] a custom dashboard was required for live monitoring, smart diagnostics, data analysis, as well as data post-processing. For this purpose, the following steps were taken: Creating a list of requirements with clearly defined functionalities. Outlining a design and user experience functionality (must have & nice to have). A schematic view of the live dashboard functionality is given in Fig. 6

The project requirements outline a comprehensive set of needs from both an operative and academic perspective. From an operative standpoint, there is a clear need for a dashboard system capable of visualizing spectroscopic data directly from the openBIS data storage and management. This includes the ability to monitor and diagnose issues with the laser instrument based on the data stored in the LIMS. Additionally, there is a requirement to execute data processing codes with special functionalities, such as compressive sensing.[46] Moreover, the dashboard should facilitate diagnostics of tabletop laser systems, enabling the evaluation of specific parameters and displaying the system status. On the user side, efficiency and functionality are key. Users need rapid uploading of spectroscopic measurements from the laser system to openBIS, with a target time of less than 10 seconds per upload. They also require tools for visualizing and interacting with the spectroscopic data, such as zooming and shifting functionalities.

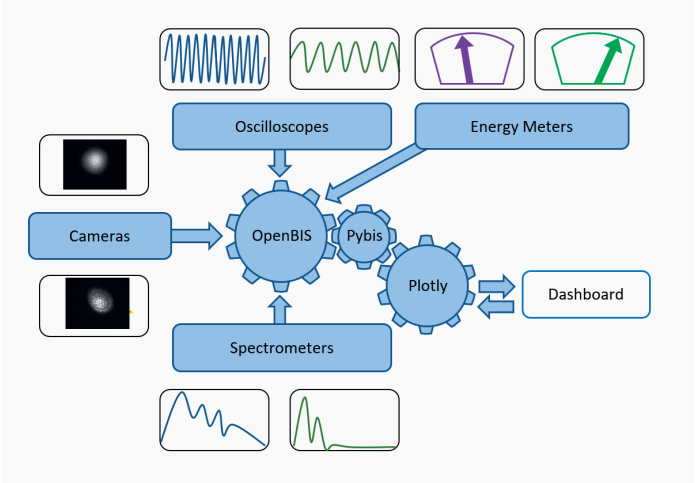

Fig. 6. Scheme of the functionality of the Laser Facility dashboard.

Furthermore, the dashboard should stream diagnostic videos from up to six cameras, allowing users to take snapshots of each camera's feed. Users should also be able to adjust camera settings like pixel clock, framerate, and exposure directly within the dashboard and ensure that snapshots are saved directly to openBIS with proper references. Moreover, users require quick access to diagnostic data from various sources. This includes diagnostic data from oscilloscopes and energy meters, which should be uploaded to openBIS within seconds. Users should also have the capability to analyze specific parameters, such as the maximum peak intensity from the digital oscilloscope. Additionally, energy meter readings should be uploaded to openBIS every 45 seconds, while the dashboard updates every 2 seconds.

Data storage and archiving also plays an important role here, as the live data is uploaded from the laser diagnostics elements at predefined intervals. Such kinds of laser facility need approximately 1 Terabyte of storage space per month.

Following the detailed delineation of these requirements, a preliminary version of the dashboard was developed and subjected to user testing. This iterative process involved gathering feedback from users to refine the dashboard's features and functionalities, ensuring it effectively meets the diverse needs outlined by both business stakeholders and end-users. After definition of the context and user requirements, a first pilot version of the dashboard was created and tested by the user. A wrapper for the API pybis package is advantageous, which is a customized set of functions which reduces the complexity for developers. Because the laser facility is equipped with multiple diagnostics and spectrometers, multiple oscilloscopes, energy meters and multiple cameras, these sensors need to be connected and configured with automatic data storage. This was configured using a Jython-dropbox similar to the one shown in the previous section. Functions to download the data to the dashboard server and display it to the user in a web-interface were developed in a generalized way to ensure reusability for developers.

It is apparent that reusable code is designed in a way that enables usage to future developers, as there are no hard-coded variables. Rather, all variables are used as inputs to the function. For visualization of the synchronized data from openBIS, the open-source library Plotly was used.[47] It provides functionalities to use templates as well as highly customized, interactive plots. Using the data downloaded according to the live data retrieval, visualizations with specific functionalities in mind were designed. Like the data retrieval, the visualizations were created in a design-pattern, which enables code to be re-used.

To fulfill the requirements on diagnostic functionality of the dashboard, the data needs to be evaluated for specific properties (*e.g.* extracting statistics, applying algorithms). Because of the highly customized nature of these processing steps in every project, we focus on some widely used functions and their implementation.

Table 4. Functions implemented for the dashboard and their description.

Function name	Description of function
fetch_diagnostic_picture_objects	Gets all datasets with type 'Diagnostics_Image' from openBISs and creates an object for each before storing it in a dictionary.
check_ramp_max_in_range	Checks if the maximum value of an array is in a given range.
calculate_fwhm	Calculates the full width at half maximum of an array
x_at_y_max_count	Gets the index of the maximal y and the full width at half maximum of an array
extract_sample_indices	Extracts indices of the list that corresponds to the reconstructed dataset.
compressive_reconstruction	This function reconstructs a signal from a sparse dataset using compressed sensing by optimizing the solution to the underdetermined linear system with l1- normalization and a conical solver.

It is important that these functions are not written in a so-called 'in-place pattern', where the original data structure is modified. In the field of computer science, the data structures are classified as mutable and immutable. The latter approach is often needed to be adapted by functional programming languages whereas compared to the former, the data structure is not modified in memory but new copies are created to retain the original in memory. A list of functions implemented for the dashboard and their respective affects are listed in the Table 4.

Lastly, it should be possible to re-upload processed data and link it to the raw data – and ideally also link it to the code with which it was processed. The newly created data will be flagged such that its source is known. This is especially important for the function which reconstructs spectra, such that the computed result can be re-used. In Fig. 7, the screenshot of the live dashboard is presented as an example.

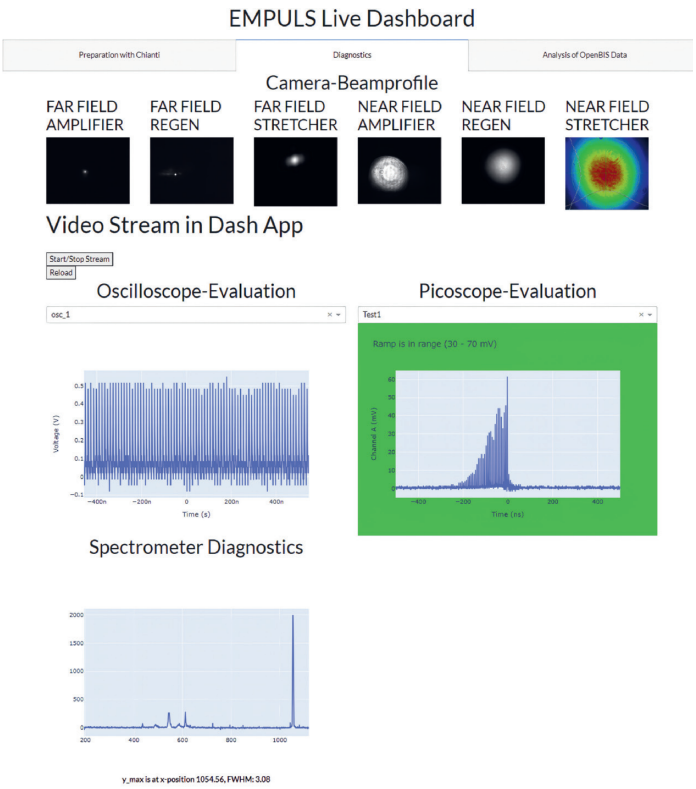

Fig. 7. A screenshot of the live dashboard for the EMPULS laser facility shows spectra, beam profiles, oscillator pulse train, amplifier ramp-up signal.

6. Conclusions

In this project, we have developed a comprehensive set of best practices for the use of openBIS. OpenBIS is an open-source ELN and LIMS, which is enhanced in combination with Python programming for digitalization projects within the scope of laboratory data management. The aim of this paper was to summarize the processes and techniques developed over a two-year period, highlighting the capabilities and ease-of-use of openBIS in modern laboratory settings. The key findings and takeaways are summarized as 10 golden rules, as follows:

1. *openBIS Complies with FAIR Data Management*: openBIS offers a flexible framework that facilitates adherence to the FAIR data management principles (Findable, Accessible, Interoperable, and Reusable). By using openBIS, researchers can efficiently manage large datasets with a focus on data quality and metadata.
2. *Understand Data Structure and Workflow Before Digitaling*: The importance of planning the data structure and workflow in advance was emphasized. It is essential to define the desired metadata, properties, and processing steps before implementing the system. This thoughtful approach ensures that data management aligns with the specific needs of the laboratory.
3. *Map Physical and Digital Structure*: We discussed the physical and digital structures of the laboratory and how openBIS can accommodate both general hierarchical structures and customized configurations to suit the needs of cross-functional teams and individual groups. The fine-grained user access management in openBIS ensures data privacy and security within a collaborative environment.
4. *Do Data Cleaning and Parsing*: Data cleaning and parsing are crucial steps in the data management process. Methods for converting structured data from external sources into openBIS objects were explored, highlighting the importance of maintaining data integrity during these processes.
5. *Have a Policy for Handling Missing Values*: Managing missing data is a common challenge when working with experimental data. It is suggested that administrators and users assess default values for objects if feasible.
6. *Shape and Mapping Properties to Objects*: The process of mapping source properties to openBIS objects is discussed, emphasizing the use of data conversion to match openBIS property notations. This step is critical for creating well-structured and organized data objects with rich metadata content on openBIS.
7. *Batch Push Datasets*: For efficient data migration, we introduced the concept of batch uploading datasets, particularly for files held in complex folder structures. The use of Python programming language simplifies the process of data transfer to openBIS.
8. *Dashboard to Pull Data/Metadata*: We presented a case study where we developed a custom dashboard for live diagnostics and analytics in a laboratory setting. We outlined the requirements, design considerations, and technical aspects of creating a dashboard tailored to specific user needs.
9. *Data Visualization is Key to Fast Insights*: In the dashboard development process, the use of open-source libraries like Plotly and Dash for data visualization and the importance of standardized design patterns to ensure code reusability was discussed.
10. *Data Traceability Requires Iteration*: We stressed the significance of re-uploading processed data into the ELN after analysis and linking it to the respective raw data to ensure traceability. By linking the processed data to its original raw data, the openBIS system maintains a clear connection between the two. Flagging newly created data helps maintain transparency and source attribution.

In conclusion, the integration of openBIS with Python and the development of custom tools and dashboards have the potential to greatly improve data management in research laboratories.

openBIS provides a robust platform for all research laboratories and is especially relevant for analytical labs as it streamlines the management of large volumes of scientific data and organization of complex datasets. Its ability to integrate with different instruments and existing workflows makes it adaptable to specific lab needs, improving efficiency and productivity. It enhances data integrity and traceability, ensuring that all experimental data and metadata are securely stored and easily accessible. It also enhances collaboration by allowing researchers to share data and track experiments in real-time while also supporting compliance with regulatory standards. These features make openBIS a valuable tool for any laboratory environment, ultimately accelerating scientific discovery and innovation.

By following the principles and techniques outlined in this paper, researchers can enhance their data management practices,

making their work more efficient and in line with contemporary standards. Ultimately, the successful digitalization of laboratory data is crucial for advancing research in various scientific disciplines.

Appendix I: List of Abbreviations

FAIR	Findable, Accessible, Identifiable, Reachable
ETL	Extract, Transform, Load
JSON	JavaScript object notation
ELN	Electronic Lab Notebook
LIMS	Laboratory Information Management System
SOP	Standard Operating Procedure
CSV	Comma separated values
IS	Information systems
MS	Mass Spectrometry
FWHM	Full Width Half Max
API	Application Programming Interface
LC	Liquid Chromatography
JS	JavaScript
RPC	Remote Procedure Call
REST	Representational State Transfer
APIs	Application programming Interface
GC	Gas Chromatography

Acknowledgements

The project is funded by SBFI within the Horizon-Europe Project EuPRAXIA-PP project in the WP7.

Author Contributions

Kilian Koch: Writing, Data Science; Yousuf Hemani: Writing, Editing, Revision, Dashboard implementation; Oscar Mendo Diaz: Editing, Mass Spectrometry Application; Anusch Bachhofner : Editing, IT Support; Simone Baffeli: Editing, IT Support; Davide Bleiner: IP Conceptualization, Editing, Work Supervision.

Received: April 24, 2024

[1] M. D. Wilkinson, M. Dumontier, Ij. 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. Dil-lo, 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 Mulli-gen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, B. Mons, *Sci Data* **2016**, *3*, https://doi.org/10.1038/sdata.2016.18.

[2] C. Barillari, D. S. M. Ottoz, J. M. Fuentes-Serna, C. Ramakrishnan, B. Rinn, F. Rudolf, *Bioinformatics* **2015**, *32*, 638, https://doi.org/10.1093/bioinformatics/btv606.

[3] S. S. Furlaneto, A. L. dos Santos, C. S. Hara, 'IEEE Network Operations and Management Symposium', **2012**, https://doi.org/10.1109/noms.2012.6211889.

[4] Z. Guan, J. Li, L. Wu, Y. Zhang, J. Wu, X. Du, *IEEE Internet Things J.* **2017**, *4*, 1934, https://doi.org/10.1109/jiot.2017.2690522.

[5] I. Segovia Ramirez, F. P. Garcia Marquez, 'Proceedings of the Fourteenth International Conference on Management Science and Engineering Management' **2020**, 470, https://doi.org/10.1007/978-3-030-49829-0_35.

[6] C.-Y. Wang, C.-C. Hsu, *Plasma Sources Sci. Technol*. **2019**, *28*, 105013, https://doi.org/10.1088/1361-6595/ab45e5.

[7] S. E. Reichenbach, *Compr. Anal. Chem*. **2009**, *77*, https://doi.org/10.1016/s0166-526x(09)05504-4.

[8] M. Ernst, D. B. Silva, R. R. Silva, R. Z. N. Vêncio, N. P. Lopes, *Nat. Prod. Rep*. **2014**, *31*, 784, https://doi.org/10.1039/c3np70086k.

[9] H. Ulrich, A.-K. Kock-Schoppenhauer, N. Deppenwiese, R. Gött, J. Kern, M. Lablans, R. W. Majeed, M. R. Stöhr, J. Stausberg, J. Varghese, M. Dugas, J. Ingenerf, *J. Med. Internet Res*. **2022**, *24*, e25440, https://doi.org/10.2196/25440.

[10] A. K. Rajasekar, R. W. Moore, 'High-Performance Computing and Networking' **2001**, 72, https://doi.org/10.1007/3-540-48228-8_8.

[11] S. G. Higgins, A. A. Nogiwa-Valdez, M. M. Stevens, *Nat. Protoc*. **2022**, *17*, 179, https://doi.org/10.1038/s41596-021-00645-8.

[12] M. Kihlén, M. Waligorski, *Drug Discovery Today* **2003**, *8*, 1007, https://doi.org/10.1016/s1359-6446(03)02887-3.

[13] H. K. Machina, D. J. Wild, *SLAS Tech*nology **2013**, *18*, 264, https://doi.org/10.1177/2211068213484471.

[14] 'LIMS: Implementation and Management', Royal Society Of Chemistry, **2007**, https://doi.org/10.1039/9781847551245.

[15] R. R. Mahaffey, 'LIMS', Springer US, **1990**, https://doi.org/10.1007/978-1-4684-9105-0.

[16] R. Kwok, *Nature* **2018**, *560*, 269, https://doi.org/10.1038/d41586-018-05895-3.

[17] D. Kolva, C. Paszko, J. Post, *Chemom. Intell. Lab. Syst*. **1994**, *26*, 171, https://doi.org/10.1016/0169-7439(94)90006-x.

[18] R. D. McDowall, *Anal. Chim. Acta* **1999**, *391*, 149, https://doi.org/10.1016/s0003-2670(99)00107-5.

[19] M. Rubacha, A. K. Rattan, S. C. Hosselet, *JALA* **2011**, *16*, 90, https://doi.org/10.1016/j.jala.2009.01.002.

[20] P. J. Prasad, G. L. Bodhe, *Chemom. Intell. Lab. Syst*. **2012**, *118*, 187, https://doi.org/10.1016/j.chemolab.2012.07.001.

[21] T. Craig, R. Holland, R. D'Amore, J. R. Johnson, H. V. McCue, A. West, V. Zulkower, H. Tekotte, Y. Cai, D. Swan, R. P. Davey, C. Hertz-Fowler, A. Hall, M. Caddick, *ACS Synth. Biol*. **2017**, *6*, 2273, https://doi.org/10.1021/acssynbio.7b00212.

[22] G. Marceddu, T. Dallavilla, A. Xhuvani, M. Daja, L. De Antoni, A. Casadei, M. Bertelli, *Acta Bio. Med. Atenei Parmensis* **2020**, *91*, e2020015, https://doi.org/10.23750/abm.v91i13-S.10521.

[23] J. Prilusky, E. Oueillet, N. Ulryck, A. Pajon, J. Bernauer, I. Krimm, S. Quevillon-Cheruel, N. Leulliot, M. Graille, D. Liger, L. Trésaugues, J. L. Sussman, J. Janin, H. van Tilbeurgh, A. Poupon, *Acta Crystallogr. Sect. D Biol. Crystallogr*. **2005**, *61*, 671, https://doi.org/10.1107/s0907444905001290.

[24] A. Melo, A. Faria-Campos, D. DeLaat, R. Keller, V. Abreu, S. Campos, *BMC Genomics* **2010**, *11*, S8, https://doi.org/10.1186/1471-2164-11-s5-s8.

[25] E. D. Foster, E. C. Whipple, G. R. Rios, *J. Med. Libr. Assoc.* **2022**, *110*, https://doi.org/10.5195/jmla.2022.1407.

[26] N. Argento, *EMBO Rep.* **2020**, 21, https://doi.org/10.15252/embr.201949862.

[27] A. Bauch, I. Adamczyk, P. Buczek, F.-J. Elmer, K. Enimanev, P. Glyzewski, M. Kohler, T. Pylak, A. Quandt, C. Ramakrishnan, C. Beisel, L. Malmström, R. Aebersold, B. Rinn, *BMC Bioinf.* **2011**, 12, https://doi.org/10.1186/1471-2105-12-468.

[28] A. Jacobsen, R. de Miranda Azevedo, N. Juty, D. Batista, S. Coles, R. Cornet, M. Courtot, M. Crosas, M. Dumontier, C. T. Evelo, C. Goble, G. Guizzardi, K. K. Hansen, A. Hasnain, K. Hettne, J. He-ringa, R. W. W. Hooft, M. Imming, K. G. Jeffery, R. Kaliyaperumal, M. G. Kersloot, C. R. Kirkpatrick, T. Kuhn, I. Labastida, B. Magagna, P. McQuilton, N. Meyers, A. Montesanti, M. van Reisen, P. Rocca-Serra, R. Pergl, S.-A. Sansone, L. O. B. da Silva Santos, J. Schneider, G. Strawn, M. Thompson, A. Waagmeester, T. Weigel, M. D. Wilkinson, E. L. Willighagen, P. Wittenburg, M. Roos, B. Mons, E. Schultes, *Data Intell.* **2020**, *2*, 10, https://doi.org/10.1162/dint_r_00024.

[29] M. Boeckhout, G. A. Zielhuis, A. L. Bredenoord, *Eur. J. Hum. Genet*. **2018**, *26*, 931, https://doi.org/10.1038/s41431-018-0160-0.

[30] H. Lütcke, 'Persistent Identifiers and the openBIS Research Data Manatement System', ETH Zurich **2019**, https://doi.org/10.3929/ETHZ-B-000365764.

[31] F. Plass, S. Englisch, B. Apeleo Zubiri, L. Pflug, E. Spiecker, M. Stingl, *CODATA* **2023**, *22*, 44, https://doi.org/10.5334/dsj-2023-044.

[32] R. El-Athman, J. Rädler, O. Löhmann, A. Ariza, T. Muth, *Proc. Conf. Res. Data Infrastr*. **2023**, 1, https://doi.org/10.52825/cordi.v1i.229.

[33] P. Simon, R. Herrmann, R. Schneider, F. Hille, M. Baeßler, R. El-Athman, 'Bridge Safety, Maintenance, Management, Life-Cycle, Resilience and Sustainability', Taylor & Francis Group, **2022**, 1061, https://doi.org/10.1201/9781003322641-127.

[34] F. G. Ringwald, A. Dudchenko, P. Knaup, F. Czernilofsky, S. Dietrich, M. Ganzinger, 'Studies in Health Technology and Informatics' *IOS Press Books* **2024**, https://doi.org/10.3233/shti231118.

[35] N. J. Knight, S. Kanza, D. Cruickshank, W. S. Brocklesby, J. G. Frey, *IEEE Internet Things J.* **2020**, *7*, 8631, https://doi.org/10.1109/jiot.2020.2995323.
[36] K. A. Badiola, C. Bird, W. S. Brocklesby, J. Casson, R. T. Chapman, S. J. Coles, J. R. Cronshaw, A. Fisher, J. G. Frey, D. Gloria, M. C. Grossel, D. B. Hibbert, N. Knight, L. K. Mapp, L. Marazzi, B. Matthews, A. Milsted, R. S. Minns, K. T. Mueller, K. Murphy, T. Parkinson, R. Quinnell, J. S. Robinson, M. N. Robertson, M. Robins, E. Springate, G. Tizzard, M. H. Todd, A. E. Williamson, C. Willoughby, E. Yang, P. M. Ylioja, *Chem. Sci*. **2015**, *6*, 1614, https://doi.org/10.1039/c4sc02128b.
[37] S. Feister, K. Cassou, S. Dann, A. Döpp, P. Gauron, A. J. Gonsalves, A. Joglekar, V. Marshall, O. Neveu, H.-P. Schlenvoigt, M. J. V. Streeter, C. A. J. Palmer, *High Power Laser Sci. Eng*. **2023**, *11*, https://doi.org/10.1017/hpl.2023.49.
[38] P. Vassiliadis, *Int. J. Data Warehousing Mining* **2009**, *5*, 1, https://doi.org/10.4018/jdwm.2009070101
[39] openBIS Documentation, User Documentation. Available at: https://openbis.readthedocs.io/en/20.10.x/index.html (Accessed: 08 April 2024).
[40] X. Chu, I. F. Ilyas, S. Krishnan, J. Wang, 'Proceedings of the 2016 International Conference on Management of Data', **2016**, https://doi.org/10.1145/2882903.2912574.
[41] R. Betancourt, S. Chen, 'Python for SAS Users', **2019**, 243, https://doi.org/10.1007/978-1-4842-5001-3_6.
[42] S. V. Chekanov, 'Advanced Information and Knowledge Processing' **2016**, 27, https://doi.org/10.1007/978-3-319-28531-3_2.
[43] Y. Hemani, M. Galimberti, D. Bleiner, 'International Conference on X-Ray Lasers 2020' **2021**, https://doi.org/10.1117/12.2595406.
[44] Y. Hemani, M. Galimberti, K. Koch, H. Panuganti, D. Bleiner, 'High-power, High-energy Lasers and Ultrafast Optical Technologies' **2023**, https://doi.org/10.1117/12.2673119.
[45] Y. Hemani, H. Panuganti, M. Galimberti, D. Bleiner, 'Compact Radiation Sources from EUV to Gamma-rays: Development and Applications' **2023**, https://doi.org/10.1117/12.2665575.
[46] Y. Hemani, K. Koch, D. Bleiner, *Spectrochimica Acta, Part B* **2024**, 106885, https://doi.org/10.1016/j.sab.2024.106885.
[47] R. Li, U. Bilal, *Biometrics* **2021**, *77*, 776, https://doi.org/10.1111/biom.13474.

License and Terms

The definitive version of this article is the electronic one that can be found at https://doi.org/10.2533/chimia.2025.36