ACS Synthetic Biology

pubs.acs.org/synthbio
Letter

The Laboratory Automation Protocol (LAP) Format and Repository: A Platform for Enhancing Workflow Efficiency in Synthetic Biology

Published as part of ACS Synthetic Biology virtual special issue "IWBDA 2022".

Ana-Mariya Anhel, Lorea Alejaldre, and Ángel Goñi-Moreno*

Cite This: *ACS Synth. Biol.* 2023, 12, 3514–3520

ABSTRACT: Laboratory automation deals with eliminating manual tasks in high-throughput protocols. It therefore plays a crucial role in allowing fast and reliable synthetic biology. However, implementing open-source automation solutions often demands experimental scientists to possess scripting skills, and even when they do, there is no standardized toolkit available for their use. To address this, we present the Laboratory Automation Protocol (LAP) Format and Repository. LAPs adhere to a standardized script-based format, enhancing end-user implementation and simplifying further development. With a modular design, LAPs can be seamlessly combined to create customized, target-specific workflows. Furthermore, all LAPs undergo experimental validation, ensuring their reliability. Detailed information is provided within each repository entry, allowing users to validate the LAPs in their own laboratory settings. We advocate for the adoption of the LAP Format and Repository as a community resource, which will continue to expand, improving the reliability and reproducibility of the automation processes.

KEYWORDS: *protocol, automation, standard, repository*

Laboratories have been striving to "reduce the manual effort in repetitive tasks as much as possible"[1] for decades now. The objective is not only to increase production rates but also to replace error-prone manual steps with robust and reliable processes, ultimately enhancing reproducibility.[2] The degree to which this goal has been accomplished varies, depending on the nature of the scientific activity. Industrial and clinical laboratories, for example, were among the earliest adopters of automation due to their requirements for reproducibility, scalability, managing large sample volumes, and high-throughput screening.[3] In contrast, fundamental research laboratories have been slower to implement automation processes,[4] and several factors may contribute to this. First, the nature of problems encountered in basic research laboratories does not always necessitate automation processes, as a significant portion of daily activities involves result observation and hypothesis generation—without production pressures. Second, molecular biology and microbiology laboratories often lack formal interdisciplinary training, particularly in the fields of robotics and automation.[5] Last, academic laboratories often lack the financial resources to access automation equipment.

The advent of synthetic biology[6] has transformed this situation. The rational engineering of predetermined functions in living systems not only requires interdisciplinary collaboration among biological sciences, computation, physics, and data science, but also entails automation throughout the design-build-test-learn (DBTL) life-cycle.[7,8] Precision and accuracy have, to some extent, replaced scalability as primary objectives, particularly in basic research laboratories. And the availability of open-source automation equipment has facilitated access to such applications. Even relatively small tasks, such as constructing and characterizing a plasmid vector, can now be effectively carried out using automation facilities.[9] Numerous tools and methods have emerged in the past decade to automate the design of genetic constructs[10,11] and establish end-to-end workflows.[12] These workflows involve collaboration among various disciplines, encompassing design, mathematical modeling, genetic engineering, and result analysis, all working toward a shared objective. Currently, there is significant room for improvement at the convergence of software, hardware, and wetware,[13] which paves the way for laboratory automation in synthetic biology laboratories, assisting in addressing future challenges.[14−16]

At the heart of open-source automation is the utilization of program scripts, which are sets of instructions that encode molecular biology protocols. These scripts serve as inputs for liquid handling robots to automate the execution of these protocols. In this context, we introduce a resource called the Laboratory Automation Protocol (LAP) Format and Repository. Its purpose is to formalize and store experimentally validated automation scripts by providing a set of standardized

Received: June 30, 2023
Revised: November 15, 2023
Accepted: November 16, 2023
Published: November 20, 2023

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520

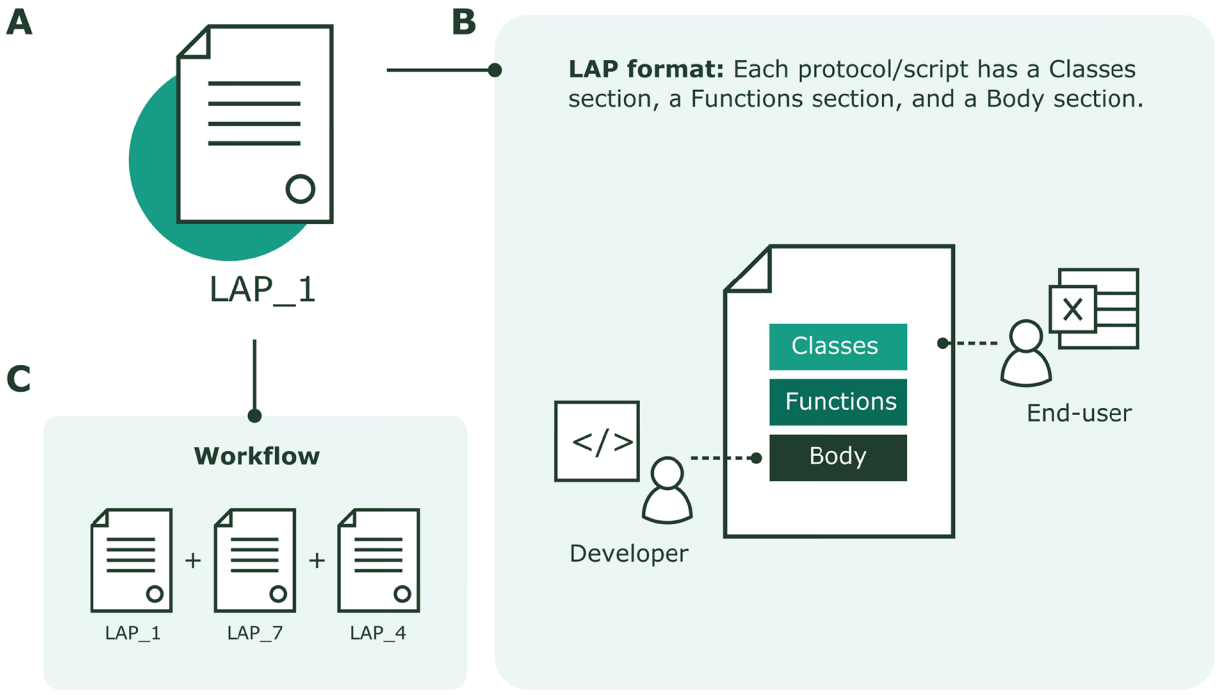

Figure 1. The LAP Format—A standard to build modular and reproducible scripts. Automation scripts are standardized, facilitating the coding of new LAPs with the aim of maximizing reusability and modularity. A. The LAP name or identifier should follow this nomenclature: repository name, brief script description, platform used, and version number, separated by hyphens. For example: LAP-ColonyCounterSelection-OT2-1.0.0. B. Each LAP consists of three sections: Classes, where users define protocol variables; Functions, which are standardized across all scripts; and Body, where developers can edit and generate protocols. C. Modularity. Various LAPs can be sequentially merged in order to run a more complex workflow.

guidelines to accelerate script development and facilitate implementation by the end-user. It is important to note that our intention is not to replace previous initiatives, but rather to complement them. For example, recent efforts in protocol-specific language description[17,18] can be used to speed up the process of defining protocol primitives. We suggest leveraging all efforts by providing a standardized structure, known as the LAP format, to improve compatibility and facilitate development. Additionally, we offer a repository to facilitate the sharing of LAP scripts and to foster collaboration among researchers. This follows a similar approach employed in other stages of the DBTL (design-build-test-learn) life cycle. For instance, in the design stage, relevant information derived from genetic construct designs, such as DNA sequences or regulatory interactions, can be efficiently captured using the SBOL[19,20] (Synthetic Biology Open Language) standard and stored in dedicated repositories like SynBioHub.[21] Another illustration is the utilization of the SEVA[22] (Standard European Vector Architecture) format and plasmid vectors during the build stage.

Figure 1 illustrates the features of the LAP Format, a standardized scripting framework. Adopting a standard format not only accelerates the creation of new protocols but also streamlines the implementation of high-throughput workflows that involve multiple sequential protocols. The proposed format entails a structured approach wherein customized variables for each entry are separated into another file, allowing users to tailor the protocol without access to the script. Given the availability of various software tools designed to generate scripts based on user specifications,[23−27] we strongly advocate for adhering to standard guidelines for script structure to promote collaboration in automated protocol development.

All protocols listed in the LAP Repository conform to this format, and any new additions must also adhere to it going forward. The file naming convention (Figure 1A) follows a specific structure: repository name (LAP), a concise description of the protocol's functionality (e.g., ColonyCounterSelection), the platform used (e.g., the OT2 liquid handling robot), and the version number (e.g., 1.0.0), separated by hyphens. For example, a protocol file could be named LAP-ColonyCounterSelection-OT2-1.0.0.

Each LAP file is divided into three sections: Classes, Functions, and Body (Figure 1B). User interaction with the protocol occurs through the Classes section, where variables specific to the protocol are defined. While values may vary, the variable names remain consistent across protocols, ensuring standardization and an easier understanding of the protocol. This interaction is facilitated by a "Customizable Variable File" that users fill in with information about the variables and the corresponding values. End-users only interact with this variable file and do not need to have any scripting background to implement open-source automation in their laboratories. The Customizable Variable File can be in various formats such as xlsx, csv, json, txt, etc. The Classes section of the LAP utilizes this file to complete the protocol specifications. The Functions section contains standardized functions that can be easily copied and pasted between different scripts depending on the requirements of the specific protocol. Users interested in developing new LAPs will likely find these functions valuable. All functions are located independently in the SetFunctions directory of the LAP GitHub repository. Any new functions developed for future LAP entries will also be added to this same folder. The Body section is where the actual protocol is coded, and its content is highly specific to each script, e.g., liquid transferring. Developers (distinct from end users) can

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520

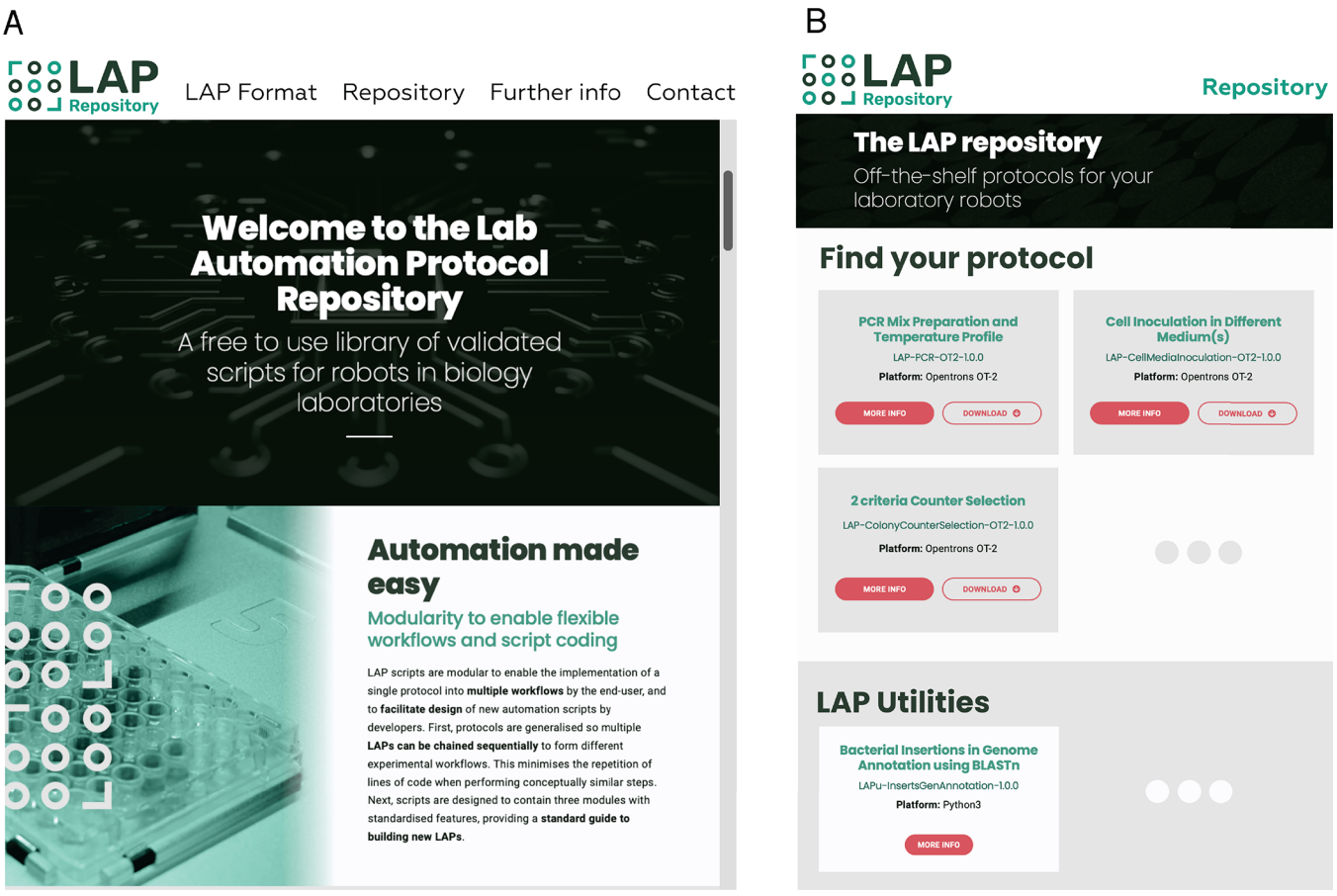

Figure 2. The LAP Repository—www.laprepo.com. Details of the home page and the Repository. A. The landing page showcases the repository, providing LAP descriptions and key features. Users can access four subpages: LAP Format, which explains the standard format; Repository, listing available LAPs; Further Info, containing technical details and information on contributing to the repository; and Contact. B. The Repository. This page provides the list of available protocols. Users can click on a specific protocol to access a dedicated page with detailed information. Additionally, the page includes a compilation of LAP Utilities, which are valuable tools for working with LAPs.

edit existing LAPs and create new ones by modifying the Body section according to their requirements. Structuring the script in this way allows for easier protocol customization, execution, and reuse of the different code blocks.

LAPs are developed with a modular approach (Figure 1C), which is crucial as it allows for the seamless combination of multiple LAPs in a sequential manner to execute more complex protocols such as workflows. For example, the LAP protocol *Cell Inoculation in Different Media* can be used as part of a workflow to genotype bacterial colonies along with LAP protocol *2 criteria Counter Selection* and *PCR Mix Preparation and Temperature Profile* as described in 10.17504/protocols.io.kqdg394jzg25/v1. LAP protocols can be reused within different workflows. Furthermore, this modularity is tailored to the end-user, as it facilitates the sharing of protocol files among different laboratories and reduces redundancy when creating new workflows.

The LAP Repository (Figure 2) is accessible through the web at www.laprepo.com. The landing page (Figure 2A) serves as an introduction to LAP and highlights our perspective on automation, emphasizing key features such as modularity and standardization. The repository is organized into four subpages: *LAP Format*, which provides an overview of the LAP Format as described in Figure 1; *Repository*, where users can explore the currently available protocols listed in the repository; *Further info*, which describes additional advantages of the LAP scripts, including the open-source nature of the project; and *Contact*, which provides contact details for the LAP Repository. Importantly, under *Further info*, there is a video explanation summarizing how users can contribute to this effort by providing LAP scripts that comply with the LAP Format, along with comprehensive descriptions of their functionality and the specific variables used for achieving the desired results. LAP curators will carefully review the submitted data before uploading it to the repository.

In the Repository subpage, which displays the list of protocols (Figure 2B), users can either browse through the available LAPs or search according to their name or function. Currently, the repository includes protocols for various purposes such as modular cloning (MoClo) assembly, sample consolidation in a single 96-well plate, 2-criteria counter selection, cell inoculation in different media, PCR mix preparation, and temperature profile. These protocols are designed to be versatile and can be applied to a wide range of experiments and workflows. To date, the LAP repository does not include the description of workflows; however, several LAPs have been used to build an example workflow available at 10.17504/protocols.io.kqdg394jzg25/v1.

Additionally, the Repository subpage features LAP Utilities (LAPu). LAP Utilities is a section that encompasses protocols or scripts that may not strictly adhere to the LAP format, such as those that do not require variables file input. However, these

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520

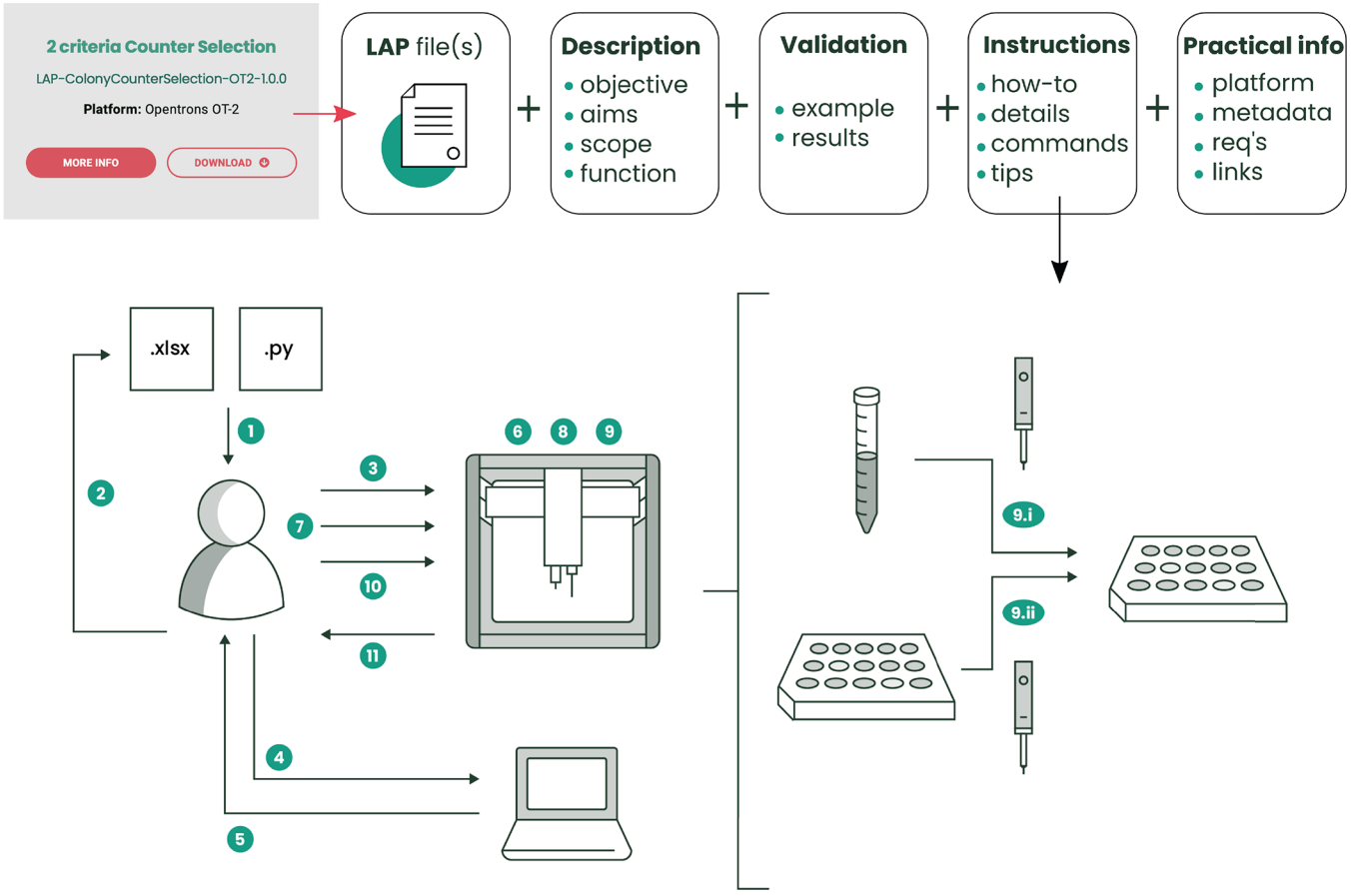

Figure 3. LAP information and description. LAPs can be expanded by clicking on the *more info* button. The description of each LAP follows a consistent layout, making it easy to comprehend. In addition to the LAP files, each protocol includes the following information: a high-level description of the protocol's objectives, an example of validation that the LAP has undergone, detailed instructions on how to run the protocol, and practical information such as the platform, code versions, requirements, and other relevant details. The step-by-step instructions are accompanied by figures to enhance understanding.

utilities play a significant role in the high-throughput dimension of automation and are utilized within workflows. Currently, there is one LAPu available for alignment and annotation using BLASTn, which is specifically used for annotating sequencing results from samples screened through an automation workflow, as showcased in 10.17504/protocols.io.kqdg394jzg25/v1.

Moving forward, all future protocols and utilities will be directly added to this list. Furthermore, new additions are not limited to a specific platform or liquid-handling robot. While the protocols listed were developed using the same platform employed in our laboratory, we strongly encourage researchers to submit protocols for other platforms as well. Also, this includes diverse methods like imaging[28] and colony picking.[29]

Users can access detailed information for each LAP by clicking on *More Info* or *Download* button. The *More Info* option presents the information in a dedicated online subpage, while the *Download* option allows users to obtain the information in a document format. LAP scripts are downloaded along with a PDF file providing instructions to run the script, an input file that corresponds to the validation described for each protocol, and a YAML file with the script's metadata. The provided information follows a standardized layout, ensuring consistency among LAPs and improving understanding. This format is expected to be followed for any new protocols added to the repository. Along with the LAP files (i.e., the LAP protocol and the Customizable Variable File), users will find a high-level description of the protocol's objectives, aims, and scope. The validation of the script for a specific use case is also included along with step-by-step instructions for running the LAP and troubleshooting potential errors. The customizable variable file downloaded with each script includes by default values used for the validation described in the *More Info* subpage for each protocol. Practical details, such as code versions and packages, the robotic platform used, and other metadata and requirements are also provided. Each LAP is accompanied by a figure (Figure 3) that visually illustrates the instructions described in the text, enhancing user comprehension.

By synergistically combining the forces of standardization and automation,[30] remarkable improvements in reproducibility and scalability throughout the DBTL life-cycle can be achieved. The LAP Format and Repository serve as a bridge in this endeavor, providing an open-source platform. This resource establishes a connection to existing and future initiatives in multifaceted ways. For instance, existing (and future) protocol generators are encouraged to embrace the LAP Format, thus boosting protocol sharing and collaboration. Furthermore, the capture and meticulous representation of protocol information[31,32] can shape the formalization of protocols at an earlier stage, providing baseline tools and abstractions to ease protocol design. Moreover, the stand-

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520

ardization efforts focused on genetic parts, exemplified by initiatives such as the SEVA Golden Standard collection[22] and the SEVA inverter package,[33] unleash their full potential within automation facilities, positioning automation protocols at the core of scalable and robust synthetic biology.

METHODS

LAP Interaction with User. Each LAP consists of two essential components. The first part is a program written in the platform's programming language that is responsible for executing the instructions required to accomplish the protocol's objectives. The second part is a customizable, structured, and human-readable file that contains various variables. Users can modify this *Customizable Variable File* (cvf) to tailor the program according to their specific experimental requirements. This file can be saved in different formats such as json, xml, or csv. The program will parse the variable file, extracting the necessary information and dynamically adapting the protocol accordingly. For the LAPs currently available (in the release version), the program format is Python, and the cvf format is xml.

One example of a Customizable Variable File would be an Excel file with multiple sheets stating the pipettes the liquid handling robot will use, the number of samples we want to handle, the volumes of each reactive, and other variables. The instructions on how to fill the variable file are set by the developer and are included in each entry of the repository.

This interaction between the script and the variable file allows the user to personalize their experimental procedures without requiring programming expertise.

The LAP Format: Classes. The LAP programs employ different classes with properties and methods to facilitate communication with the script's body and functions. These classes play a crucial role in parsing the variables provided in the CVF, performing error checking, and generating user-friendly error messages related to possible variable errors, such as trying to transfer to a lot more volume than its maximum. An important feature is that these enable accurate calculation of required volumes based on the user's variables, thus maximizing efficiency.

Common classes are shared across different protocols. For instance, the *UserVariables* class is responsible for reading, parsing, and validating the CVF provided by the user. It ensures that the values are correctly formatted and checks for potential errors. The *SetParameters* class then takes the object created by *UserVariables* and performs calculations to fill various objects or variables required by the script's body. This process allows for efficient parameter handling and preparation.

LAPs can include specific classes tailored to unique protocol functionalities. For example, the *MapLabware* class creates a data frame with the dimensions of a given labware, fills it with values such as sample names, and finally exports it to a csv file that the user can import from the liquid handling robot. This feature is highly beneficial in tracking colony samples throughout an experimental workflow, providing traceability during the experimental procedure.

While the content of these classes may vary depending on the specific variables required for a given protocol, it is encouraged to adopt consistent naming for similar variable concepts. For instance, the right-mounted pipet can always be referred to as *pipR*. This approach facilitates class translation between different LAP programs and enhances the code readability and maintenance for developers. In addition, some functions can require objects that have properties with specific names and values to be used.

The LAP Format: Functions. The LAP protocols encompass a variety of functions essential for executing the script's core tasks. These functions can be categorized into different types based on their specificity.

First, some functions are widely shared among protocols due to their utility in liquid-handling scripts. For example, the *give_me_optimal_pipette* function that determines the most suitable pipette to minimize the movements required for transferring a specific volume from one location to another. Shared standard functions ensure the codebase's modularity, reusability, and readability.

Second, some functions are specific to the individual protocols and tailored to address the unique requirements of a particular experiment. For instance, *distribute_z_tracking_falcon15 mL* performs distribution to a specified list of positions while tracking the liquid level to prevent the pipette from getting wet, and it is tailored to be used in falcons of 15 mL, so it is not recommended to be used in an experiment that will use a 50 mL falcon to hold reagents.

Both types of functions require name consistency inside of the previously discussed classes, emphasizing the importance of consistent naming conventions for variables of the same type. An example of such a function is *check_tip_and_pick* which verifies whether a given pipette has a tip attached and, if not, establishes a tip rack so the pipette can pick a tip. For this purpose, the function needs the *UserVariable* class to have a property called *APINameTipR* and *APINameTipL* that will contain the name of the tipracks that the pipettes need.

The LAP Format: Body. Every LAP protocol has a unique script body that utilizes previously established classes and functions to accomplish specific goals. This personalized method enables each LAP protocol to be customized and adjusted to meet the exact needs of an experiment.

LAP Utilities. The LAP utilities (LAPu) are intended to streamline the workflow associated with data analysis and provide researchers with efficient tools for processing and extracting information from their high-throughput experimental data. While they may not be specifically tailored for robotic liquid handling operations, they offer valuable functionalities to researchers involved in acquiring and processing large volumes of data.

The different functionalities and platform of usage make the entries in the LAP Utilities section deviate from the LAP structure and their set of classes and functions.

Validation. All LAPs underwent experimental validation. For each protocol, both input and output files from the validation process are provided, serving as a reference for verifying the correct functionality of the LAP. Furthermore, within each LAP, there is a dedicated section called *Validation* that offers additional experimental information. This includes details such as the specific reagents and conditions used during the original validation, ensuring reproducibility. The simulated labware setup and specified reagent quantities are also outlined. Therefore, to validate an LAP in a new laboratory, it is sufficient to utilize the default input files provided, along with the information presented in the validation section.

Availability. The LAP Repository can be accessed at the following web address: www.www.laprepo.com. Furthermore, all LAPs and related code are hosted in a dedicated GitHub

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520

repository, which can be found at: https://github.com/BiocomputationLab/LAPrepository.

AUTHOR INFORMATION

Corresponding Author

Ángel Goñi-Moreno − *Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid, 28223 Madrid, Spain;* orcid.org/0000-0002-2097-2507; Email: angel.goni@upm.es

Authors

Ana-Mariya Anhel − *Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid, 28223 Madrid, Spain;* orcid.org/0000-0002-4097-3638

Lorea Alejaldre − *Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid, 28223 Madrid, Spain;* orcid.org/0000-0003-3086-5446

Complete contact information is available at: https://pubs.acs.org/10.1021/acssynbio.3c00397

Author Contributions

A.G.M. conceived the study. A.M.A. and L.A. developed LAP protocols and format and validated LAPs experimentally. All the authors contributed to the discussion of the research and writing of both manuscript and Web site content.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We thank Dr. Laura Cutugno for valuable inputs during the development of LAP-MoCloAssembly-OT2-1.0.0. This work was supported by the grants BioSinT-CM (Y2020/TCS-6555) and CONTEXT (Atracción de Talento Program; 2019-T1/BIO-14053) Projects of the Comunidad de Madrid, MULTI-SYSBIO (PID2020-117205GA-I00) and the Severo Ochoa Program for Centres of Excellence in R&D (CEX2020-000999-S) funded by MCIN/AEI/10.13039/501100011033, and the ECCO (ERC-2021-COG-101044360) contract of the EU.

REFERENCES

(1) Liscouski, J. G. Laboratory automation. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 288−292.

(2) Baker, M. 1500 scientists lift the lid on reproducibility. *Nature* **2016**, *533*, 452.

(3) Chapman, T. Lab automation and robotics: Automation on the move. *Nature* **2003**, *421*, 661−663.

(4) Kitney, R.; Adeogun, M.; Fujishima, Y.; Goñi-Moreno, Á.; Johnson, R.; Maxon, M.; Steedman, S.; Ward, S.; Winickoff, D.; Philp, J. Enabling the advanced bioeconomy through public policy supporting biofoundries and engineering biology. *Trends Biotechnol.* **2019**, *37*, 917−920.

(5) Hallinan, J. S.; Wipat, A.; Kitney, R.; Woods, S.; Taylor, K.; Goñi-Moreno, A. Future-proofing synthetic biology: educating the next generation. *Engineering Biology* **2019**, *3*, 25−31.

(6) Amos, M.; Goñi-Moreno, A. Cellular computing and synthetic biology. *Computational matter* **2018**, 93−110.

(7) Carbonell, P.; Radivojevic, T.; Garcia Martin, H. Opportunities at the intersection of synthetic biology, machine learning, and automation. *ACS Synth Biol.* **2019**, *8*, 1474 DOI: 10.1021/acssynbio.8b00540.

(8) Tellechea-Luzardo, J.; Otero-Muras, I.; Goñi-Moreno, A.; Carbonell, P. Fast biofoundries: coping with the challenges of biomanufacturing. *Trends Biotechnol.* **2022**, *40*, 831.

(9) Alejaldre, L.; Anhel, A.-M.; Goñi-Moreno, A. pBLAM1-x: Standardized transposon tools for high-throughput screening. *Synthetic Biology* **2023**, *8*, ysad012.

(10) Densmore, D. M.; Bhatia, S. Bio-design automation: software+ biology+ robots. *Trends Biotechnol.* **2014**, *32*, 111−113.

(11) Appleton, E.; Densmore, D.; Madsen, C.; Roehner, N. Needs and opportunities in bio-design automation: four areas for focus. *Curr. Opin. Chem. Biol.* **2017**, *40*, 111−118.

(12) Goñi-Moreno, A.; Carcajona, M.; Kim, J.; Martinez-Garcia, E.; Amos, M.; de Lorenzo, V. An implementation-focused bio/algorithmic workflow for synthetic biology. *ACS synthetic biology* **2016**, *5*, 1127−1135.

(13) Oliveira, S. M. D.; Densmore, D. Hardware, Software, and Wetware Codesign Environment for Synthetic Biology. *BioDesign Research* **2022**, *2022*, 2022.

(14) Gallup, O.; Ming, H.; Ellis, T. Ten future challenges for synthetic biology. *Engineering Biology* **2021**, *5*, 51−59.

(15) Grozinger, L.; Amos, M.; Gorochowski, T. E.; Carbonell, P.; Oyarzún, D. A.; Stoof, R.; Fellermann, H.; Zuliani, P.; Tas, H.; Goñi-Moreno, A. Pathways to cellular supremacy in biocomputing. *Nat. Commun.* **2019**, *10*, 5250.

(16) Hanson, A. D.; Lorenzo, V. d. Synthetic Biology High Time to Deliver? *ACS Synth. Biol.* **2023**, *12*, 1579−1582.

(17) Wierenga, R.; Golas, S.; Ho, W.; Coley, C.; Esvelt, K. PyLabRobot: An Open-Source, Hardware Agnostic Interface for Liquid-Handling Robots and Accessories. *bioRxiv* **2023**, *1*, 100111.

(18) Bartley, B.; Beal, J.; Rogers, M.; Bryce, D.; Goldman, R. P.; Keller, B.; Lee, P.; Biggers, V.; Nowak, J.; Weston, M. Building an Open Representation for Biological Protocols. *ACM Journal on Emerging Technologies in Computing Systems* **2023**, *19*, 1−21.

(19) Madsen, C.; Goni Moreno, A.; P, U.; Palchick, Z.; Roehner, N.; Atallah, C.; Bartley, B.; Choi, K.; Cox, R. S.; Gorochowski, T.; Grunberg, R.; Macklin, C.; McLaughlin, J.; Meng, X.; Nguyen, T.; Pocock, M.; Samineni, M.; Scott-Brown, J.; Tarter, Y.; Zhang, M.; Zhang, Z.; Zundel, Z.; Beal, J.; Bissell, M.; Clancy, K.; Gennari, J. H.; Misirli, G.; Myers, C.; Oberortner, E.; Sauro, H.; Wipat, A. Synthetic biology open language (SBOL) version 2.3. *Journal of integrative bioinformatics* **2019**, *16*, 20190025.

(20) Crowther, M.; Grozinger, L.; Pocock, M.; Taylor, C. P.; McLaughlin, J. A.; Mısırlı, G.; Bartley, B. A.; Beal, J.; Goñi-Moreno, A.; Wipat, A. ShortBOL: a language for scripting designs for engineered biological systems using Synthetic Biology Open Language (SBOL). *ACS synthetic biology* **2020**, *9*, 962−966.

(21) McLaughlin, J. A.; Myers, C. J.; Zundel, Z.; Mısırlı, G.; Zhang, M.; Ofiteru, I. D.; Goni-Moreno, A.; Wipat, A. SynBioHub: a standards-enabled design repository for synthetic biology. *ACS synthetic biology* **2018**, *7*, 682−688.

(22) Martinez-Garcia, E.; Fraile, S.; Algar, E.; Aparicio, T.; Velazquez, E.; Calles, B.; Tas, H.; Blazquez, B.; Martin, B.; Prieto, C.; Sanchez-Sampedro, L.; Nørholm, M. H H; Volke, D. C; Wirth, N. T; Dvorak, P.; Alejaldre, L.; Grozinger, L.; Crowther, M.; Goni-Moreno, A.; Nikel, P. I; Nogales, J.; de Lorenzo, V.; et al. SEVA 4.0: an update of the Standard European Vector Architecture database for advanced analysis and programming of bacterial phenotypes. *Nucleic Acids Res.* **2023**, *51*, D1558−D1567.

(23) Herisson, J.; Duigou, T.; du Lac, M.; Bazi-Kabbaj, K.; Sabeti Azad, M.; Buldum, G.; Telle, O.; El Moubayed, Y.; Carbonell, P.; Swainston, N.; Zulkower, V.; Kushwaha, M.; Baldwin, G. S.; Faulon, J.-L. The automated Galaxy-SynBioCAD pipeline for synthetic biology design and engineering. *Nat. Commun.* **2022**, *13*, 5082.

(24) Whitehead, E.; Rudolf, F.; Kaltenbach, H.-M.; Stelling, J. Automated planning enables complex protocols on liquid-handling robots. *ACS synthetic biology* **2018**, *7*, 922−932.

(25) Linshiz, G.; Stawski, N.; Poust, S.; Bi, C.; Keasling, J. D.; Hillson, N. J. PaR-PaR laboratory automation platform. *ACS synthetic biology* **2013**, *2*, 216−222.

(26) Bates, M.; Berliner, A. J.; Lachoff, J.; Jaschke, P. R.; Groban, E. S. Wet lab accelerator: a web-based application democratizing

laboratory automation for synthetic biology. *ACS synthetic biology* **2017**, *6*, 167−171.
(27) Tas, H.; Grozinger, L.; Goñi-Moreno, A.; de Lorenzo, V. Automated design and implementation of a NOR gate in Pseudomonas putida. *Synthetic Biology* **2021**, *6*, ysab024.
(28) Ouyang, W.; Bowman, R. W.; Wang, H.; Bumke, K. E.; Collins, J. T.; Spjuth, O.; Carreras-Puigvert, J.; Diederich, B. An Open-Source Modular Framework for Automated Pipetting and Imaging Applications. *Advanced biology* **2022**, *6*, 2101063.
(29) del Olmo Lianes, I.; Yubero, P.; Gomez-Luengo, A.; Nogales, J.; Espeso, D. R. Technical upgrade of an open-source liquid handler to support bacterial colony screening. *bioRxiv* **2023**, *2023−02*, 530165.
(30) Beal, J.; Goni-Moreno, A.; Myers, C.; Hecht, A.; de Vicente, M. d. C.; Parco, M.; Schmidt, M.; Timmis, K.; Baldwin, G.; Friedrichs, S.; Freemont, P.; Kiga, D.; Ordozgoiti, E.; Rennig, M.; Rios, L.; Tanner, K.; de Lorenzo, V.; Porcar, M. The long journey towards standards for engineering biosystems: Are the Molecular Biology and the Biotech communities ready to standardise? *EMBO reports* **2020**, *21*, No. e50521.
(31) Beal, J.; Weiss, R.; Densmore, D.; Adler, A.; Appleton, E.; Babb, J.; Bhatia, S.; Davidsohn, N.; Haddock, T.; Loyall, J.; et al. An end-to-end workflow for engineering of biological networks from high-level specifications. *ACS synthetic biology* **2012**, *1*, 317−331.
(32) Crowther, M.; Wipat, A.; Goñi-Moreno, Á. A network approach to genetic circuit designs. *ACS Synth. Biol.* **2022**, *11*, 3058−3066.
(33) Tas, H.; Goñi-Moreno, Á.; Lorenzo, V. d. A standardized inverter package borne by broad host range plasmids for genetic circuit design in Gram-negative bacteria. *ACS synthetic biology* **2021**, *10*, 213−217.

https://doi.org/10.1021/acssynbio.3c00397
ACS Synth. Biol. 2023, 12, 3514−3520