ACS Synthetic Biology

pubs.acs.org/synthbio

Technical Note

SynBioSuite: A Tool for Improving the Workflow for Genetic Design and Modeling

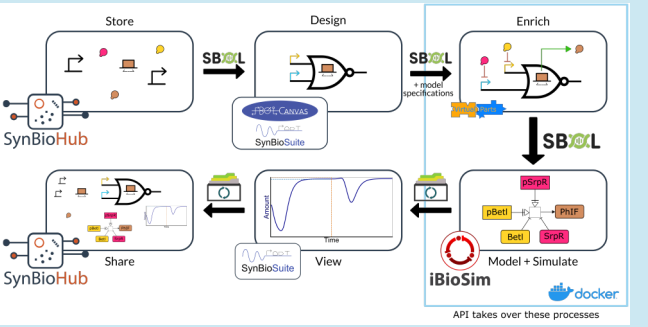
Zachary Sents, Thomas E. Stoughton, Lukas Buecherl, Payton J. Thomas, Pedro Fontanarrosa, and Chris J. Myers*

Cite This: *ACS Synth. Biol.* 2023, 12, 892–897

Read Online

ACCESS | Metrics & More | Article Recommendations

ABSTRACT: Synthetic biology research has led to the development of many software tools for designing, constructing, editing, simulating, and sharing genetic parts and circuits. Among these tools are SBOLCanvas, iBioSim, and SynBioHub, which can be used in conjunction to create a genetic circuit design following the design−build−test−learn process. However, although automation works within these tools, most of these software tools are not integrated, and the process of transferring information between them is a very manual, error-prone process. To address this problem, this work automates some of these processes and presents SynBioSuite, a cloud-based tool that eliminates many of the drawbacks of the current approach by automating the setup and reception of results for simulating a designed genetic circuit via an application programming interface.



KEYWORDS: *genetic design automation, synthetic biology, API, web server, SBOLCanvas, iBioSim*

INTRODUCTION

Synthetic biology is a field of research that involves the application of engineering principles to the design of biological systems.[1] This field has enabled many applications, including the development of biofuels,[2] internal drug delivery systems,[3] and biosensors,[4] all of which rely on genetic circuits to work. Among other methods, genetic circuits execute their functions using genetic expression regulation, metabolic pathways, or protein interactions.[5]

Research in synthetic biology—particularly the development of genetic circuits—usually follows a *design−build−test−learn* (DBTL) cycle to progress from a desired function to a physical build.[6] To aid in the DBTL cycle of genetic circuits, computational models are used to simulate the design before it is built *in vivo*.[7] This saves time and resources for researchers and can help to predict malfunctions in the circuit beforehand.[8]

Genetic design automation (GDA) tools have been developed to support the DBTL cycle by automating many of its steps, ultimately simplifying genetic circuit engineering.[6,9] This project utilizes SBOLCanvas,[10] iBioSim,[11−13] and SynBioHub[14] to create a seamless genetic design workflow. SBOLCanvas is a web application that can be used to create and annotate genetic constructs using an icon-based drag-and-drop GUI (https://sbolcanvas.org/). SBOLCanvas is capable of retrieving specific genetic parts or full designs from SynBioHub and exporting new designs that can be uploaded to SynBioHub. The iBioSim application is a downloadable software tool that can be used to design, model, and simulate genetic circuits using a variety of algorithms for ordinary differential equation or stochastic simulation analysis. Finally, SynBioHub is an online repository that enables researchers to store and share information about their genetic designs.

A set of standard description languages are used to link these tools together. These standards allow for coherent communication between tools and reproducibility of results. The languages used by the aforementioned software are the *Synthetic Biology Open Language* (SBOL),[15−17] *SBOL Visual*,[18] the *Systems Biology Markup Language* (SBML),[19] and the *Simulation Experiment Description Markup Language* (SED-ML).[20] SBOL, SBML, and SED-ML are all XML-based formats, each of which has a different purpose for the design and simulation of genetic circuits: SBOL standardizes the *in silico* representation of biological designs; SBOL Visual represents the design using glyphs; SBML files encode computational models of biological systems; and SED-ML files encode simulation descriptions for the purpose of

Received: November 7, 2022
Published: March 8, 2023

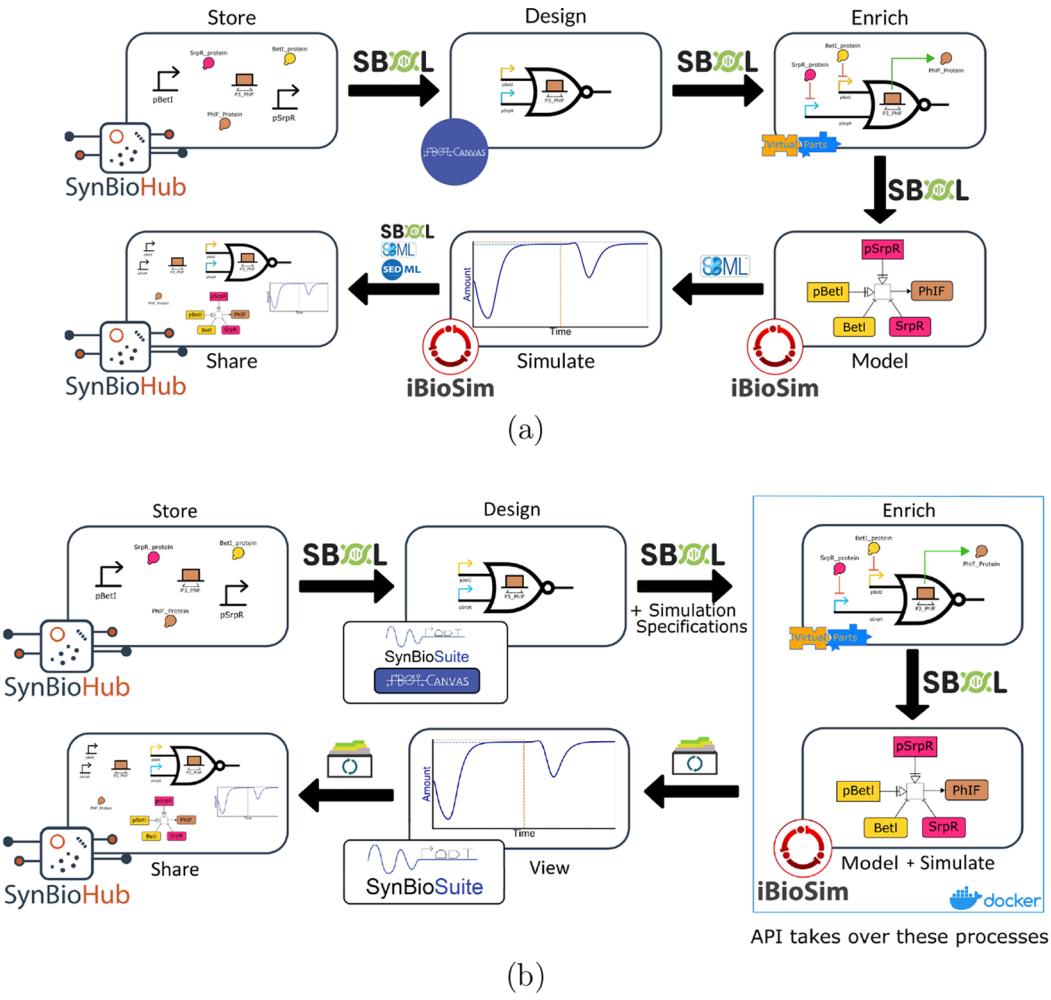


Figure 1. (a) Diagram of the previous workflow for designing and simulating a genetic circuit. This workflow involves three different applications: Selected parts stored in SynBioHub are used in the design created in SBOLCanvas. The enrichment, model, and simulate steps happen within iBioSim, and the share step happens again in SynBioHub. At each change of application, the relevant SBOL, SBML, or SED-ML files must be exported from the previous application and then imported into the next application. (b) Diagram of the new workflow. This workflow only requires the user to work in the SynBioSuite application; the enrichment, model, and simulate steps are handled in the containerized iBioSim API instance, and the simulation results are rendered back on SynBioSuite. The final design is uploaded and shared on SynBioHub. This workflow also makes use of the COMBINE archive to package the SBOL, SBML, and SED-ML files together for ease of use and reproducibility.

repeatability. In addition to these standard formats, a COMBINE archive[21] is used to package SBOL, SBML, and SED-ML files together so that an entire simulation study can be exchanged easily.

With the development of these software tools and standards, a workflow for the design, modeling, and simulation of genetic circuits is established. The previous workflow for using these software tools is specified in Figure 1a. This process of designing and testing genetic circuits starts with the design of a genetic circuit using SBOLCanvas. This tool can be used to retrieve genetic parts from SynBioHub directly. Then the design is exported in SBOL format, which can be imported into iBioSim. iBioSim uses the *Virtual Parts Repository* (VPR) model generator[22] to enrich the design with additional information about proteins and small molecules that are known to interact with the genetic parts. Then a computational model is created in iBioSim, where the SBOL file automatically gets converted to SBML.[8,23−25] Simulation conditions are specified by the user, and the simulation is run. Finally, the results of the simulation are plotted and sent back to SynBioHub for sharing.

This workflow is a complex, manual process of importing and exporting files between applications, so automation is desirable. Furthermore, while iBioSim is a useful application that is able to run complex circuit simulations, it has some drawbacks. It needs to be downloaded on the user's local machine and depends on a Java installation. In addition, its graphical user interface is outdated and may not be intuitive for new users. This paper introduces SynBioSuite (https://synbiosuite.org), a web application that seeks to aggregate the capabilities of each aforementioned tool within a convenient interface. This is accomplished by modifying SBOLCanvas to act as an embeddable service and by exposing some of the modeling and simulation capabilities of iBioSim through an *application programmers interface* (API).

RESULTS

SynBioSuite is a web application built with React, a JavaScript front-end framework. It allows the user to open a folder on their local file system, which is used as the working directory. Existing SBOL files, SBML files, and COMBINE archives will appear in the application's file explorer. When SBOL files are

https://doi.org/10.1021/acssynbio.2c00597
ACS Synth. Biol. 2023, 12, 892−897

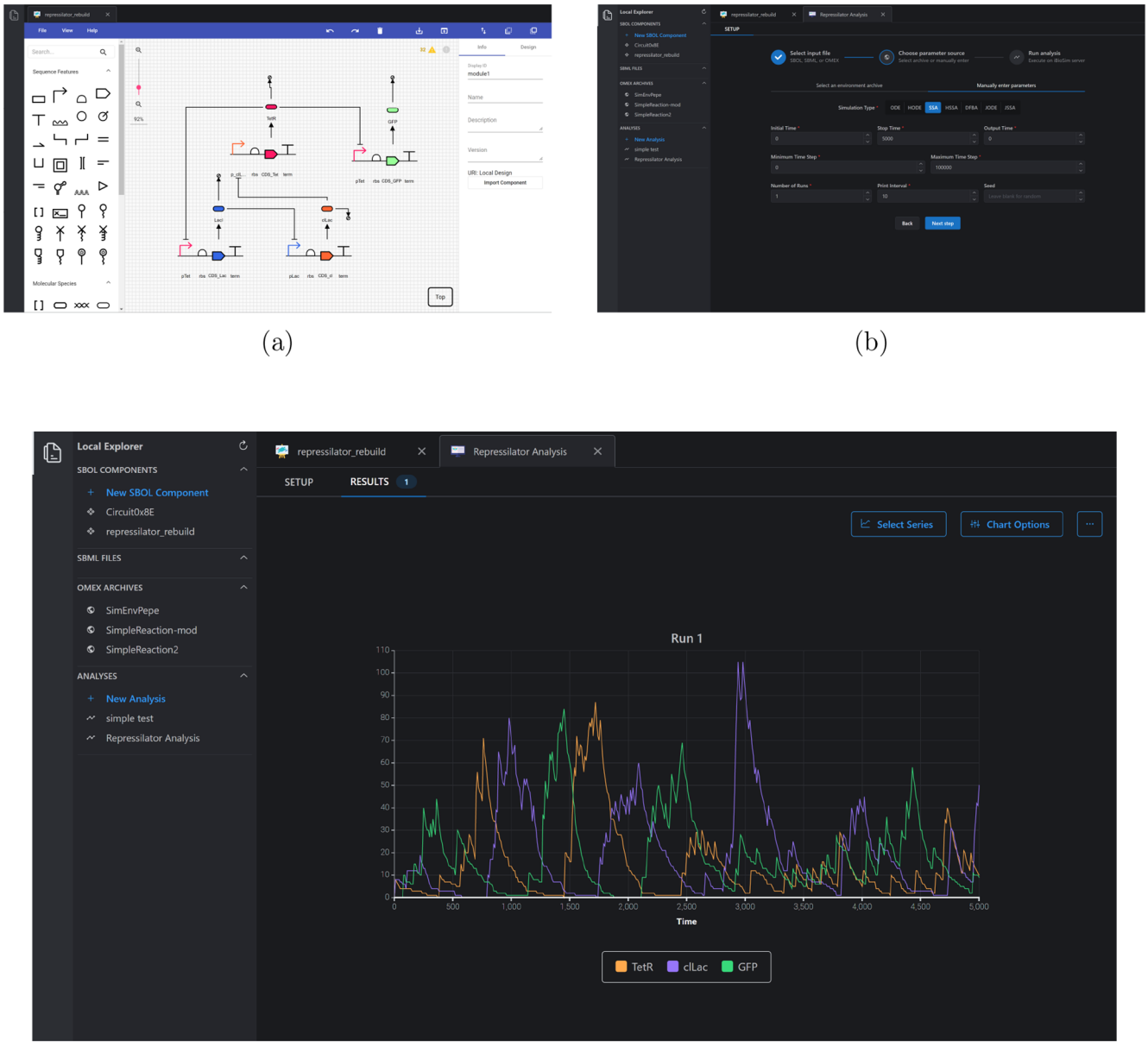


Figure 2. Screenshots from the SynBioSuite web application. (a) SBOLCanvas embedded in SynBioSuite is used to design the repressilator.[26] (b) The analysis tab allows the selection of different simulation methods and their parameters. (c) The results of the analysis are visualized as a graph. As expected, the repressilator oscillates.

double-clicked, SBOLCanvas is opened in an embedded frame with the SBOL file's contents loaded in. New SBOL files can also be created. From inside SBOLCanvas, the user can modify the design. Changes are automatically saved to the user's local file system. To simulate designs, users can create an "analysis", which opens a form where the user can choose a design to simulate, optionally add a simulation environment, and specify simulation parameters. SynBioSuite invokes the iBioSim API, which computes the results and sends them back. The results are plotted in an interactive editor where the user can modify and export the resulting charts. The current workflow is described in Figure 1b. This architecture eliminates the need for researchers to download an application as well as switch between applications. A single web application provides a more seamless workflow compared to the previous workflow. Furthermore, with each tool being a modular service, they can be leveraged individually in future work.

In order to take advantage of the tool's simulation capabilities from the SynBioSuite web app, iBioSim has been placed inside a Docker container (a virtual environment that stores an application and its dependencies for standalone execution). Alongside iBioSim is an Express application running on Node.js (a JavaScript runtime) that handles HTTP requests, which functions as the API server. The API exposes two end points, *convert* and *analyze*. The *convert* end point takes an SBOL file, along with some conversion parameters, and returns an SBML file. The *analyze* end point takes either an SBOL file, SBML file, or COMBINE archive

https://doi.org/10.1021/acssynbio.2c00597
ACS Synth. Biol. 2023, 12, 892−897

along with parameters for the enrich, model, and simulate steps. If an SBOL file is provided, it is first converted to SBML. Then the application invokes iBioSim via the command line, running the simulation. The results are packaged into a COMBINE archive and sent back to SynBioSuite. This containerized service is what is referred to as the iBioSim API.

The SBOLCanvas web application is a combination of a front end built with the JavaScript framework Angular and a back end built in Java. SBOLCanvas' front end provides an interface for designing genetic circuits with the SBOL Visual standard. The back end handles assembly of the SBOL file that describes the design as well as a few other utility tasks like interacting with SynBioHub. The application was modified to be useful when embedded in SynBioSuite. When embedded, it initially loads in the SBOL sent by the parent window (SynBioSuite). Then, as the user works, it relays changes to the design back to the parent window. Thus, the design stays in sync between the two applications. SynBioSuite's user interface can be seen in Figure 2.

Figure 2a shows SBOLCanvas embedded in SynBioSuite. The user is able to draw their design on the canvas, importing and exporting parts or whole designs to SynBioHub. In this example, the user has designed the repressilator.[26] After the design is finished, the user can set the parameters for different simulation methods in the analysis tab, shown in Figure 2b. SynBioSuite offers both ordinary differential equation analysis and stochastic simulation algorithms. For the example at hand, stochastic simulation was used to simulate the circuit. Finally, Figure 2c shows the results of the analysis, which is the expected oscillation of the repressilator.

DISCUSSION

This work describes SynBioSuite, a web application comprising several microservices, which together simplify the genetic circuit design process. This application and its constituents realize the new, streamlined workflow for virtually designing, modeling, and simulating genetic circuits. Web-based applications such as SBOLCanvas have many advantages for both users and developers over standard, downloadable applications. Some of these advantages include accessibility, ease of maintenance, and the elimination of installation issues. Furthermore, the API will always be up to date on iBioSim updates due to the nature of the Docker container, which uses the newest published version of iBioSim upon building.

This approach would allow researchers to have a single, entirely online workspace for designing and simulating genetic circuits, something that has only previously been available in downloadable application packages. Creating a new interface that taps into iBioSim's sophisticated modeling and simulation capabilities saves much effort compared to a full overhaul of the simulation tool.

With this approach, the core functionality of iBioSim is maintained, that is, the simulation and conversion of genetic circuits and models. However, there are some features that are still only accessible from iBioSim's GUI, such as the ability to create model environments in SBML using its schematic editor. In the future, SynBioSuite's GUI and the iBioSim API can be extended to restore this functionality. Plans for a plugin interface are also being discussed, which would allow external developers to augment SynBioSuite with new functionality.

There are a few limitations that exist in the SynBioSuite application. The biggest limitation is browser compatibility. Currently, SynBioSuite is only supported on Google Chrome >86 and Opera >72. This is the case because the application relies heavily on the File System Access API. The File System Access API is a browser API that provides access to a user's local file system, allowing web applications to read and write files on the user's device. This API is used to build more advanced and native-like web applications that can interact with the file system, rather than being limited to reading and writing data solely in the browser's storage or through server interactions. Currently, a version of SynBioSuite is in development that allows the application to run with limited functionality in other browsers. Although this alleviates the issue slightly, the functionality remaining without file system access makes the application much less useful. In the future, a more full-featured solution would involve access to remote file systems or Git repositories.

Furthermore, as already mentioned, the full capabilities of iBioSim are not fully represented. Also, the user is required to open a local directory before creating any designs, while users may want to simply create a new design and download it. This functionality is currently only available in the standalone SBOLCanvas application (https://sbolcanvas.org). Lastly, the application does not remember recently opened directories, which would be a convenient feature for future versions.

METHODS

SynBioSuite was built with React, a JavaScript front-end framework, and Vite, a build tool. The built application is deployed with an Azure service called Static Web Apps. This service leverages global content delivery networks (CDNs) to serve static web applications. The SBOLCanvas front end is deployed similarly.

The iBioSim API consists of two components inside a Docker container: iBioSim (a Java application) and an API server built with the Express library running with Node.js. The API is deployed with an Azure service called Container Apps, which is a serverless implementation for running containerized applications. Container Apps have advantages over standard virtual machines because they only incur charges while processing data and they automatically scale horizontally. Azure Container Apps also support revisions, which are essentially snapshots of the application, its container images, and configuration. The service automatically generates a new revision each time that changes to the underlying container image are made, so the deployed service is always up to date. The Java-based SBOLCanvas back end is deployed similarly.

One caveat with long-running processes like simulations via HTTP requests is that the client needs to remain open to receive the response. This is problematic because a user on SynBioSuite will lose any simulation progress if the page refreshes or is closed. To circumvent this, a middleman was implemented using an Azure service called Durable Functions (DF). DF leverages serverless execution and cloud storage to keep track of state for long-running operations. When SynBioSuite sends a request to DF, the request is forwarded to the iBioSim API, but DF responds immediately with a URL that SynBioSuite can use to check the status of the operation. This way, SynBioSuite can simply store that URL locally and check back in on the simulation whenever the page is reopened.

https://doi.org/10.1021/acssynbio.2c00597
ACS Synth. Biol. 2023, 12, 892−897

■ ASSOCIATED CONTENT

Data Availability Statement

SynBioSuite can be accessed at https://synbiosuite.org. The source code is available at https://github.com/MyersResearchGroup/SynBioSuite. The repository provides four examples with instructions to help with using SynBioSuite for the first time. The code of the API is open-source, available on https://github.com/MyersResearchGroup/iBioSim-API, and released with the Apache 2.0 License. Documentation is available in the README in the GitHub repository. The source code for SBOLCanvas can be accessed at https://github.com/SynBioDex/SBOLCanvas/tree/synbio-suite. The latest version of iBioSim, including source code, instructions, and related files, can be found online at https://github.com/MyersResearchGroup/iBioSim.

■ AUTHOR INFORMATION

Corresponding Author

Chris J. Myers − *Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, Colorado 80309, United States;* orcid.org/0000-0002-8762-8444; Email: chris.myers@colorado.edu

Authors

Zachary Sents − *Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, Colorado 80309, United States;* orcid.org/0000-0002-9467-3534

Thomas E. Stoughton − *Department of Computer Science, University of Colorado Boulder, Boulder, Colorado 80309, United States*

Lukas Buecherl − *Biomedical Engineering Program, University of Colorado Boulder, Boulder, Colorado 80309, United States;* orcid.org/0000-0002-4844-6605

Payton J. Thomas − *Department of Biomedical Engineering, University of Utah, Salt Lake City, Utah 84112, United States*

Pedro Fontanarrosa − *Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, Colorado 80309, United States;* orcid.org/0000-0002-0535-2684

Complete contact information is available at: https://pubs.acs.org/10.1021/acssynbio.2c00597

Author Contributions

Z.S. and C.J.M. developed SynBioSuite and the most recent version of the iBioSim API. T.E.S. and C.J.M. developed and tested the first version of the iBioSim API. L.B. mentored Z.S. and T.E.S. throughout the project and added a dockerized plugin of the API for SynBioHub allowing simulation access directly from SynBioHub. P.F. developed the conversion and parts of the analysis functionality of iBioSim. P.J.T. provided a dockerized image of iBioSim's conversion and analysis code base. All of the authors contributed to the writing of the paper and created all figures in their entirety and all images used in the TOC graphic.

Funding

The authors of this work were supported by DARPA Grant FA8750-17-C-0229, National Science Foundation Grant 1856740, and the University of Colorado Boulder Engineering Excellence Fund. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The authors thank the members of the Genetic Logic Lab at the University of Colorado Boulder and the University of Utah (https://www.geneticlogiclab.org) for their feedback on the software and the manuscript.

■ REFERENCES

(1) Cheng, A. A.; Lu, T. K. Synthetic Biology: An Emerging Engineering Discipline. *Annu. Rev. Biomed. Eng.* **2012**, *14*, 155−178. PMID: 22577777.

(2) Olson, D. G.; McBride, J. E.; Shaw, A. J.; Lynd, L. R. Recent progress in consolidated bioprocessing. *Curr. Opin. Biotechnol.* **2012**, *23*, 396−405.

(3) Shi, P.; Gustafson, J. A.; MacKay, J. A. Genetically engineered nanocarriers for drug delivery. *Int. J. Nanomed.* **2014**, *9*, 1617.

(4) Vigneshvar, S.; Sudhakumari, C.; Senthilkumaran, B.; Prakash, H. Recent advances in biosensor technology for potential applications-an overview. *Front. Bioeng. Biotechnol.* **2016**, *4*, 11.

(5) Brophy, J. A. N.; Voigt, C. A. Principles of Genetic Circuit Design. *Nat. Methods* **2014**, *11*, 508−520.

(6) Xiang, Y.; Dalchau, N.; Wang, B. Scaling up genetic circuit design for cellular computing: advances and prospects. *Nat. Comput.* **2018**, *17*, 833−853.

(7) Schlitt, T. Approaches to modeling gene regulatory networks: A gentle introduction. *Methods Mol. Biol.* **2013**, *1021*, 13−35.

(8) Fontanarrosa, P.; Doosthosseini, H.; Borujeni, A. E.; Dorfan, Y.; Voigt, C. A.; Myers, C. J. Genetic Circuit Dynamics: Hazard and Glitch Analysis. *ACS Synth. Biol.* **2020**, *9*, 2324−2338.

(9) Buecherl, L.; Myers, C. J. Engineering Genetic Circuits: Advancements in Genetic Design Automation Tools and Standards for Synthetic Biology. *Curr. Opin. Microbiol.* **2022**, *68*, 102155.

(10) Terry, L.; Earl, J.; Thayer, S.; Bridge, S.; Myers, C. J. SBOLCanvas: A Visual Editor for Genetic Designs. *ACS Synth. Biol.* **2021**, *10*, 1792−1796.

(11) Watanabe, L.; Nguyen, T.; Zhang, M.; Zundel, Z.; Zhang, Z.; Madsen, C.; Roehner, N.; Myers, C. IBIOSIM 3: A Tool for Model-Based Genetic Circuit Design. *ACS Synth. Biol.* **2019**, *8*, 1560−1563.

(12) Myers, C. J.; Barker, N.; Jones, K.; Kuwahara, H.; Madsen, C.; Nguyen, N.-P. D. iBioSim: A Tool for the Analysis and Design of Genetic Circuits. *Bioinformatics* **2009**, *25*, 2848−2849.

(13) Roehner, N.; Myers, C. J. Directed Acyclic Graph-Based Technology Mapping of Genetic Circuit Models. *ACS Synth. Biol.* **2014**, *3*, 543−555.

(14) 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 Synth. Biol.* **2018**, *7*, 682−688.

(15) Galdzicki, M.; Clancy, K. P.; Oberortner, E.; Pocock, M.; Quinn, J. Y.; Rodriguez, C. A.; Roehner, N.; Wilson, M. L.; Adam, L.; Anderson, J. C.; et al. The Synthetic Biology Open Language (SBOL) provides a community standard for communicating designs in synthetic biology. *Nat. Biotechnol.* **2014**, *32*, 545−550.

(16) Roehner, N.; Beal, J.; Clancy, K.; Bartley, B.; Misirli, G.; Grunberg, R.; Oberortner, E.; Pocock, M.; Bissell, M.; Madsen, C.; et al. Sharing structure and function in biological design with SBOL 2.0. *ACS Synth. Biol.* **2016**, *5*, 498−506.

(17) McLaughlin, J. A.; Beal, J.; Mısırlı, G.; Grünberg, R.; Bartley, B. A.; Scott-Brown, J.; Vaidyanathan, P.; Fontanarrosa, P.; Oberortner, E.; Wipat, A.; et al. The Synthetic Biology Open Language (SBOL) version 3: simplified data exchange for bioengineering. *Front. Bioeng. Biotechnol.* **2020**, *8*, 1009.

(18) Baig, H.; Fontanarossa, P.; Kulkarni, V.; McLaughlin, J.; Vaidyanathan, P.; Bartley, B.; Bhakta, S.; Bhatia, S.; Bissell, M.; Clancy, K.; et al. Synthetic Biology Open Language Visual (SBOL Visual) Version 2.3. *J. Integr. Bioinf.* **2021**, *18*, 20200045.
(19) Keating, S. M.; Waltemath, D.; König, M.; Zhang, F.; Dräger, A.; Chaouiya, C.; Bergmann, F. T.; Finney, A.; Gillespie, C. S.; Helikar, T.; et al. SBML Level 3: an extensible format for the exchange and reuse of biological models. *Mol. Syst. Biol.* **2020**, *16*, e9110.
(20) Waltemath, D.; Adams, R.; Bergmann, F. T.; Hucka, M.; Kolpakov, F.; Miller, A. K.; Moraru, I. I.; Nickerson, D.; Sahle, S.; Snoep, J. L.; et al. Reproducible computational biology experiments with SED-ML—the simulation experiment description markup language. *BMC Syst. Biol.* **2011**, *5*, 198.
(21) Bergmann, F. T.; Adams, R.; Moodie, S.; Cooper, J.; Glont, M.; Golebiewski, M.; Hucka, M.; Laibe, C.; Miller, A. K.; Nickerson, D. P.; et al. COMBINE archive and OMEX format: one file to share all information to reproduce a modeling project. *BMC Bioinf.* **2014**, *15*, 369.
(22) Mısırlı, G.; Yang, B.; James, K.; Wipat, A. Virtual Parts Repository 2: Model-driven design of genetic regulatory circuits. *ACS Synth. Biol.* **2021**, *10*, 3304−3315.
(23) Roehner, N.; Zhang, Z.; Nguyen, T.; Myers, C. J. Generating systems biology markup language models from the synthetic biology open language. *ACS Synth. Biol.* **2015**, *4*, 873−879.
(24) Nguyen, T.; Roehner, N.; Zundel, Z.; Myers, C. J. A converter from the systems biology markup language to the synthetic biology open language. *ACS Synth. Biol.* **2016**, *5*, 479−486.
(25) Misirli, G.; Nguyen, T.; McLaughlin, J. A.; Vaidyanathan, P.; Jones, T. S.; Densmore, D.; Myers, C.; Wipat, A. A computational workflow for the automated generation of models of genetic designs. *ACS Synth. Biol.* **2019**, *8*, 1548−1559.
(26) Elowitz, M. B.; Leibler, S. A synthetic oscillatory network of transcriptional regulators. *Nature* **2000**, *403*, 335−338.

https://doi.org/10.1021/acssynbio.2c00597
ACS Synth. Biol. 2023, 12, 892−897