Status and Challenges of Reproducibility in Computational Systems and Synthetic Biology

Chapter · January 2019

DOI: 10.1016/B978-0-12-801238-3.11525-9

Kiri Choi
Yale University
49 PUBLICATIONS **494** CITATIONS

Herbert M Sauro
University of Washington
303 PUBLICATIONS **14,471** CITATIONS

Status and Challenges of Reproducibility in Computational Systems and Synthetic Biology

Kiri Choi[1,2], Jonathan R. Karr[3], and Herbert M. Sauro[2]

[1]Korea Institute for Advanced Study, Seoul, Korea
[2]Department of Bioengineering, University of Washington, Seattle, WA, USA
[3]Icahn Institute and Department of Genetics and Genomic Sciences, Icahn School of Medicine at Mount Sinai, New York, NY, USA

Abstract

Scientific research is reproducible when the findings can be independently verified. Reproducibility is crucial for the integrity of science. Unfortunately, scientific studies, including computational studies, are often not reproducible. We believe there are two primary causes for the frequent lack of reproducibility of computational systems and synthetic biology studies. First, the information needed to reproduce a result is often not communicated clearly. This issue can be addressed by improving and expanding the existing standards, the support for the standards, and the communication of the standards. Second, the computational environment needed to reproduce a result is often not shared. This issue can be partly addressed with virtual machines. Here, we outline the status of the reproducibility of computational systems and synthetic biology by reviewing the existing standards and software tools. As part of this review, we highlight some of the most common standards and software tools. Additionally, we discuss the shortcomings of the current standards and software tools, highlighting several gaps which continue to make computational systems and synthetic biology studies challenging to reproduce. In particular, we highlight the need for expanded standards for describing the provenance and verification of computational systems biology models.

Keywords— annotation, COMBINE archive, domain-specific language, reproducibility, SBML, SBOL, SED-ML, standards, synthetic biology, systems biology, workflow

1 Introduction

Reproducibility is a cornerstone of scientific research. Unfortunately, over the past several years, several studies have demonstrated that scientific results are often not reproducible [1–3]. This has led researchers and funding agencies to call for more reproducible research [1]. Many experimental biomedical studies are inherently challenging to reproduce because it is often difficult to control every experimental variable. For example, it is challenging to reproduce mammalian cell culture experiments that rely on undefined media derived from animal sources.

Here, we focus on the reproducibility of computational systems and synthetic biology. At first glance, computational studies are expected to be reproducible because computations can be precisely controlled, unlike wet laboratory experiments. However, computational studies are often irreproducible. This irreproducibility is often due to incomplete, untested, outdated, or missing instructions for running the software; missing, outdated, or unannotated source code files; or missing or incorrect data. Without well-annotated source code, readers must try to reproduce computational experiments from scratch, relying on the descriptions provided in publications, which are often incomplete and prone to errors. Furthermore, many times computational results cannot be independently verified via distinct computational or experimental methods. This is because the assumptions made in computational experiments are often not communicated clearly.

The poor reproducibility of computational systems and synthetic biology studies is a significant problem. As computational systems and synthetic biology studies become more ambitious, requiring a higher degree of collaboration, such as building comprehensive models of entire cells, tissues, and organs, reproducibility will become critical to establish trust in each other's work. The concept of model reuse, which allows the construction of a multi-scale model by merging several smaller sub-models, requires reproducibility. Here, we review recent efforts to increase the reproducibility of computational systems and synthetic biology, highlighting the most common standards and software tools for reproducible research in these fields. Finally, we discuss the limitations of the current approaches and the outlook for increasing the reproducibility of computational systems and synthetic biology.

2 Overview of Past Efforts to Make Computational Systems and Synthetic Biology Reproducible

Computational experiments can be independently reproduced through several different means [4]. One approach is to containerize entire computational experiments using technologies such as Docker containers [5]. The benefit of this approach is that one can exchange the entire computing environment with others, making the approach the most reproducible out of all the options. However, this approach requires technical expertise that many biological labs do not have. It is also inefficient and inflexible in the sense that a container contains a lot of unnecessary information and is hard to modify. Another approach is to use workflows [6]. Workflows are generally concise, editable, and easy to use. The downside is workflows are difficult to preserve over the long term because they do not encapsulate the software used in workflows [7]. A third approach is to define domain-specific standards for describing models and experiments and develop software tools to support these standards. One of the primary advantages of standards is its longevity. For example, zip files created in 1990 can still be uncompressed today because modern compression utilities continue to support the zip standard that was developed in 1989 [8]. Unfortunately, developing and establishing standards requires community agreement and extensive effort from both the researchers and software developers.

Different computational biology communities have developed standards to exchange different types of information. The original objective of these standards was to exchange models, simulation experiments, and designs between multiple software tools without losing information. As a result, the standards help make computational studies largely reproducible. In particular, several standards have been developed for the systems and synthetic biology over the last 20 years. Here, we discuss four of the most common standards, including the Systems Biology Markup Language (SBML) [9], the Simulation Experiment Description Markup Language (SED-ML) [10], the COMBINE archive [11], and the Synthetic Biology Open Language (SBOL) [12, 13]. For a detailed description of each standard, we suggest you check other chapters dedicated to standards.

2.1 Standards for systems and synthetic biology

The Systems Biology Markup Language (SBML) [9] describes models of biochemical processes such as cell signaling pathways and metabolism. The core SBML standard describes kinetic models. Additionally, there are currently twelve SBML packages which extend SBML to describe additional types and aspects of models. These packages range from supporting random distributions and constraint-based models to hierarchical compositions and layouts for a visual representation of pathways. As a *de facto* standard, SBML is widely used by the community. For example, the BioModels model repository (`https://www.ebi.ac.uk/biomodels`) uses SBML to represent models.

CellML [14] is also a popular standard for describing computational systems biology models. In contrast to SBML, which emphasizes the semantic biological meaning of models, CellML focuses on the mathematics of models. This focus on mathematics makes CellML applicable to a broader range of scales – tissues, organs, and whole organisms – than SBML. However, the focus on mathematics also makes CellML-encoded models harder to comprehend, reuse, and compose.

The Simulation Experiment Description Markup Language (SED-ML) describes simulation experiments [10]. This includes the algorithm(s) used in the experiments, the parameters for each simulation run, which simulation prediction should be recorded, and how these predictions should be analyzed and plotted. SED-ML is necessary because model descriptions are insufficient to reproduce simulation results. Together, SBML and SED-ML can capture enough information to completely reproduce a simulation result such as a figure in an article.

The COMBINE archive is a standard for archiving all of the information necessary to reproduce a computational experiment. This includes raw data, model descriptions (e.g., SBML files), simulation experiment descriptions (e.g., SED-ML files), and more [11]. In particular, the COMBINE archive aims to standardize the encapsulation of the files needed to reproduce a computational study into a single, easily transportable file.

The Synthetic Biology Open Language (SBOL) describes genetic designs for synthetic organisms [13, 15], including information about the genetic sequence, the components, and their interactions. SBOL also supports hierarchical designs to enable researchers to collaboratively compose genetic elements into entire synthetic organisms.

Standards have also been developed to facilitate collaboration on other aspects of computational systems and synthetic biology such as the semantic meaning, graphical depiction, and revision history of models and genetic designs. For example, the Systems Biology Ontology (SBO) is a vocabulary for describing the semantic meaning of a model [16]. COMODI is an ontology for describing the change history of a model [17]. The Systems Biology Result Markup Language (SBRML) [18] is a format for describing the numerical results of a computational experiment. The Systems Biology Graphical Notation (SBGN) [19] and SBOL Visual [20] are standards for describing visual depictions of models and designs.

2.2 Standards for other biomodeling domains

There are also other standards for other areas of computational biology. For example, NeuroML [21] is a standard for describing computational neuroscience models. pharmML [22] is a standard for describing pharmacometrics models. Norm-sys (`https://normsys.h-its.org`) outlines several additional standards for computational biology. The Open Biological and Biomedical Ontology (OBO) Foundry [23] (`http://www.obofoundry.org`) and BioPortal (`https://bioportal.bioontology.org`) outline numerous useful vocabularies for annotating biomodeling studies.

Now that we examined the most common standards used in several biomedical modeling communities, we now discuss several tools for using these standards. We also highlight several successful scientific projects that have been enabled by these standards and software tools.

3 Current State of Software Support for Reproducible Study

Using the standards described above requires user-friendly, standards-compliant software tools for building and simulating models. Many software applications for systems biology support SBML to varying degrees including, but not limited to, CellDesigner [24], COPASI [25], iBioSim [26], Jarnac [27], JWS Online [28], openCOR [29], PathwayDesigner [30] (formerly called JDesigner), PySCeS [31], SBW [32], Tellurium [33], and TinkerCell [34]. Overall, there is more than 290 software currently available that supports SBML and the list is constantly growing. This widespread support has been spurred by the availability of libraries for reading, writing, and modifying SBML-encoded models for C, C++, Java, MATLAB, Python, and other languages. The list of software that supports SED-ML and the COMBINE archive is smaller but several of the software tools listed here also support SED-ML, and in some cases, the COMBINE archive, allowing the users to import and/or export simulation experiments along with the model. The example includes CellDesigner, COPASI, iBioSim, JWS Online, openCOR, and Tellurium. It is not necessary for software that supports SBML to support SED-ML and the COMBINE. Many of them are not for simulation and analysis but model design, visualization, etc. However, we would like to see more simulation software to support SED-ML and the COMBINE archive as the information encoded in these standards are crucial for reproducibility of computational studies.

Software	**Descriptions**
CellDesigner	Diagram-based editor that also supports simulation and analysis (`http://www.celldesigner.org`)
COPASI	GUI-based simulation and analysis tool for biochemical networks (`http://copasi.org`)
iBioSim	JAVA-based simulation and analysis tool targeted for genetic circuits (`http://www.async.ece.utah.edu/ibiosim`)
Jarnac	Simulator for reaction networks (`http://systems-biology.org/software/simulation/jarnac.html`)
JWS Online	Web-based tool for modeling and simulation (`http://jjj.biochem.sun.ac.za`)
openCOR	GUI-based simulation and analysis platform for CellML (`http://www.opencor.ws`)
PathwayDesigner	GUI-based graphical modeling environment for biochemical networks (`http://pathwaydesigner.org`)
PySCeS	Python package for simulation and analysis (`http://pysces.sourceforge.net`)
SBOLDesigner	GUI-based CAD software tool for designing genetic circuits that supports SBOL (`http://www.async.ece.utah.edu/SBOLDesigner`)
SBW	Software framework integrating various applications for systems biology (`http://sbw.sourceforge.net`)
Tellurium	Python-based modeling and simulation environment (`http://tellurium.analogmachine.org`)
TinkerCell	GUI-based CAD software for genetic circuits (`http://www.tinkercell.com`)

Table 1: Software discussed in this chapter.

Although SBOL is a comparatively new standard, SBOL gained support from several software tools including iBioSim, SBOLDesigner [35], Tellurium, and TinkerCell, which allows export/import of SBOL 1/2 files and sometimes even Genbank and FASTA files. Online repositories such as SynBioHub [36] are available for supporting SBOL. Similar to SBML libraries, SBOL libraries are available for multiple languages such as C++, Java, JavaScript, and Python leading to wider adoption. Table 1 lists a tiny fraction of the many tools which support standards discussed here. The SBML (`https://sbml.org`) and SBOL websites (`http://sbolstandard.org`) maintain extensive lists of tools which support SBML and SBOL. Together, these software tools support a wide range of tasks including designing models (e.g., PathwayDesigner, Tellurium), simulating models (e.g., COPASI, JWS online, Tellurium), and visualizing models (e.g., CellDesigner).

4 Success Stories of the Impact of Reproducible Research

As computational systems and synthetic biology become more reproducible, we expect computational systems and synthetic biology to become more impactful and more collaborative. Availability of exchangeable studies through repositories as well as software that support standards have allowed researchers to expand and explore new ideas by analyzing and reusing models [37–39].

In terms of individual subjects, cardiac electrophysiology is a field which has benefited from reproducible researches. The cardiac electrophysiology field can use the Chaste [40] library, which supports CellML, to simulate models reproducibly. The field has also used Chaste to develop the Cardiac Electrophysiology Web Lab [41] which enables researchers to reproducibly simulate models via a user-friendly web-based interface. Together, Chaste and the Cardiac Electrophysiology Web Lab have enabled new studies [42] and encouraged the development of other software tools for reproducible cardiac modeling [43, 44]. Similarly, liver metabolism modeling has benefited from reproducible and exchangeable studies. Systems such as HepatoNet1 model [45] and HMR2.0 database [46] are encoded in SBML, allowing researchers from diverse fields to utilize it for new developments and validations [47–50]. Outside of systems biology, nervous system modeling, specifically those involving *Caenorhabditis elegans* has been made possible from reproducible studies. OpenWorm project [51] has been utilizing NeuroML extensively, which have provided the groundwork for new research [52] and software [53].

Recently, the Center for Reproducible Biomedical Modeling has been organized, led by the collaboration of various researchers from institutions across the world (`https://reproduciblebiomodels.org/`). The center aims to build and provide assistant to designing multi-scale models by servicing reproducible workflow and expanding standards in terms of both the specification and software.

5 Open Challenges in Reproducing Computational Systems and Synthetic Biology Studies

Despite the standards and software tools described above, computational systems and synthetic biology studies remain difficult to reproduce. As a result, we believe that these standards must be expanded to make computational systems and synthetic biology fully reproducible. Here, we outline several open obstacles to the reproducibility of computational systems and synthetic biology studies and propose potential solutions to these issues through new and expanded standards.

5.1 Reproducing computational studies involving numerous tasks

Computational biology studies often involve numerous tasks, software tools, and standards. We believe that reproducing such complex studies requires adopting or developing new standards and new tools.

Although SED-ML was designed to capture complex simulation studies, we believe that SED-ML has some issues that necessitate an alternative approach. First, SED-ML cannot describe complex pre and post-processing. For example, SED-ML cannot describe metabolic control analysis, bootstrapping, or the generation of histograms. This not only severely limits the application of SED-ML, but also increases the maintenance. Second, it is difficult to programmatically generate SED-ML descriptions of simulation studies because it is hard to translate computational procedures into SED-ML's declarative representation. Third, it is difficult to read and write SED-ML because the format is not human-readable and only a few software tools support SED-ML while improving the readability. This make it difficult for researchers to use SED-ML to clearly communicate simulation experiments.

Attempts have been made to improve the readability [54] but are limited to a small subset of environments. Taken together, we believe that SED-ML must be substantially revised to support the community's needs to clearly communicate complex simulation studies.

A domain-specific programming language such as SESSL [55] could be a good alternative, or even an extension, to SED-ML for describing complex simulation studies. A domain-specific language is a language for a specific domain, unlike a general-purpose language that applies to many domains. SED-ML is a domain-specific markup language, not a domain-specific programming language. First, a domain-specific programming language would make it easier to implement complex simulation experiments through a scripting language. The syntax can support a wide range of functionalities present in modern programming languages to support various pre and post processing done to data such as array manipulation and iterations. Second, a domain-specific programming language could be much more flexible than SED-ML, supporting custom extensions. A domain-specific programming language can support defining custom functions and modularization. Third, a domain-specific programming language can be much more readable and writable. A domain-specific programming language can be designed to be a high-level programming language akin to Python, MATLAB, and Julia. The standard can be naturally readable without necessitating a translator or a visualization software. While there are many domain-specific programming languages, there are few suitable languages for computational biomedical studies. Even SESSL is not completely suitable because the language is implemented only in Scala, making it inadequate for widespread adoption.

Workflow management systems are another potential alternative to SED-ML for describing complex simulation studies. Workflow management systems make it easy to construct, describe, and execute complex workflows that involve multiple steps, tools, and formats [56]. First, workflow management systems help users build complex workflows by providing graphical or other high-level interfaces for designing workflows. Second, some workflow management systems help users collaboratively develop workflows by sharing workflows through central databases. Third, many workflow management systems increase the comprehensibility of workflows by abstracting tasks and the flow of information among tasks. Fourth, some workflow management systems help users quickly execute workflows by identifying parallelism in workflows and help users use clusters or cloud resources to run independent tasks in parallel. Fifth, some workflow management systems record the intermediate state in workflows to help users reproduce and restart workflows. Currently, there are over 250 workflow management systems for a wide range of purposes and applications. Some of the most popular workflow management systems in computational biology include Galaxy [57], Kepler [58], and Taverna [59]. Also, the Common Workflow Language [60] (CWL) and the myExperiment workflow repository [61] enable researchers to exchange workflows between management systems. However, as of May 2019, only a few systems supported the CWL.

We recommend the computational systems biology community adopt CWL as a standard for describing modeling workflows and execute workflows with CWL-compatible engines such as Cromwell and Toil [62]. Otherwise, workflows described in proprietary formats will become difficult to reproducible once the format is no longer supported by a workflow engine. To make these technologies even easier to use, we recommend that the computational systems biology community also develop tools for designing workflows for computational systems biology and an ontology for describing the semantic meaning of tasks encoded in CWL. Furthermore, we recommend the adoption of a container management system such as DockerHub to ensure that the software needed to execute individual tasks is available in perpetuity.

Both domain-specific programming language and workflow approaches have different use cases. A domain-specific language would be best suited for workloads which can be accomplished within a single computational environment, such as MATLAB, Python, and R. In contrast, workflows are best-suited to workloads which require multiple tools.

5.2 Increasing the usability of standards

The systems biology community has been mostly focusing on improving the exchangeability of studies, that is, the ability to exchange information over many people while ensuring behavior to be conserved over different environments. However, Exchangeability is not the only feature necessary to ensure reproducibility. Transparency, which is the ability to communicate the content of a study in an understandable manner, is another aspect of a model that is crucial for reproducibility [63]. For this reason, we believe that standards should be easy to understand to help reproducibility. Consequently, we recommend that new standards be designed in human-readable/writable formats that can be viewed and edited without requiring special software. Since existing standards are not designed in this way, we suggest software developers to at least include tools to help the readability of the standards, similar to what Antimony [64] and phraSED-ML [54] are providing.

To transparently communicate studies, other types of information must be communicated as well. These are annotation, which provides information on what is being studied, and provenance, which provides information on why a study was implemented in such a way, which are discussed next.

5.3 Concretely communicating the semantic meaning of models

While SBML supports annotations, SBML often does not concretely capture the semantic meaning of a model. For example, because SBML relies on textual ids, identifiers in external namespaces such as UniProt, and ontologies such as Gene Ontology (GO) [65] to describe the semantic meaning of variables that represent proteins, and because these namespaces do not represent each possible variable of each protein, SBML is unable to concretely represent the semantic meaning of the protein variants represented by models. As a concrete example, SBML does not provide a mechanism to concretely describe the meaning of a variable that represents a specific phosphorylated form of MAPK, including the residues which are phosphorylated and the chemical structures of the phosphorylations. Although such meta-information may not be needed to simulate the model, this information is important to understand the model, and this information may be vital to verify the model such as checking that the model is consistent with mass conservation. A new format is being developed to annotate CellML and SBML models [66]. We believe the new format should concretely describe the DNA, RNA, and proteins represented by models. The proposed new format will be included in COMBINE archives.

5.4 Reproducing model construction and verification

While systems biology has devoted considerable attention to reproducing simulation results, the community has devoted less attention to reproducing the construction and verification of models. For example, SBML was not designed to capture structured information about the assumptions and design decisions needed to build a model, or the simulations needed to verify a model. Git logs and the COMODI ontology [17] can capture the development history of a model such as the sequence of changes that were made, who made each change, and why each change was made; the PROV-O [67] and Scientific Evidence and Provenance Information Ontology (SEPIO) [68] ontologies provide generic models for provenance information; and MEMOTE [69], NeuroUnit [70], and SciUnit [71] have begun to explore formats for describing how to verify models. However, the construction and verification of systems biology models are often irreproducible, in part, because the systems biology community has not yet embraced these tools, and in part, because the available standards do not capture all information necessary to reproduce a study.

We believe that new standards are needed to capture the provenance and verification of studies. The provenance standard should capture information about the data, assumptions, and design decisions needed to build a model, in addition to the change history of the model. This standard should also be flexible to accommodate the different ways that researchers may want to record provenance. Furthermore, the standard should be accompanied by guidelines about the minimum provenance information that should be communicated with a computational study. The verification standard should capture the simulations and analyses necessary to verify a model, and the data needed to evaluate each prediction, as well as the confidence in each model prediction and the range of input parameters that can be trusted. Additionally, a comprehensive framework for verifying systems biology models should be developed. Potentially, this could be implemented by expanding existing frameworks such as pytest and existing formats for test results such as the XUnit XML format. Both standards should be encapsulated into COMBINE archives.

6 Conclusion

The systems and synthetic biology communities have made significant progress toward understanding, reproducing, and reusing computational systems and synthetic biology studies. However, several obstacles remain to reproduce computational systems and synthetic biology studies. In particular, new and expanded standards and tools are needed to reproduce the construction, significance, calibration, and verification of models.

Open standards and software are one of the keys to reproducibility, but there are also significant cultural barriers that must be overcome. We believe that the current reward system and scientific culture should be shifted to encourage researchers to publish reproducible science. Ideally, we believe that funding agencies would insist that published work be reproducible. However, this would be difficult to enforce, and this heavy-handed approach would likely have the side effect of discouraging innovation. Instead, we recommend that journals work with dedicated verification services to verify the reproducibility of results before publication. For example, the *Journal of Political Science* (`https://ajps.org/ajps-replication-policy`) works with the Odum Institute for Research in Social Science to verify the reproducibility of submitted results, and the journal only publishes papers with verified results. Furthermore, funding agencies could insist that publications be published in journals that verify results. Despite these challenges, we are optimistic that we can make computational systems and synthetic biology more reproducible through a combination of new standards, new software tools, and increased emphasis on the importance of reproducibility.

7 Acknowledgements

This work was supported by National Institutes of Health award P41 EB023912. KC was supported by NIH award R01-GM123032. The content is solely the responsibility of the author and does not necessarily represent the views of the National Institutes of Health.

References

1. Baker, M. 1,500 scientists lift the lid on reproducibility. *Nature* **533,** 452–454 (2016).
2. Begley, C. G. & Ellis, L. M. Drug development: Raise standards for preclinical cancer research. *Nature* **483,** 531–533 (2012).
3. Prinz, F., Schlange, T. & Asadullah, K. Believe it or not: how much can we rely on published data on potential drug targets? *Nat Rev Drug Discov* **10,** 712–712 (2011).
4. Barba, L. A. Terminologies for reproducible research. *arXiv preprint arXiv:1802.03311* (2018).
5. Howe, B. Virtual appliances, cloud computing, and reproducible research. *Computing in Science & Engineering* **14,** 36–41 (2012).
6. Garijo, D., Kinnings, S., Xie, L., *et al.* Quantifying reproducibility in computational biology: the case of the tuberculosis drugome. *PloS one* **8,** e80278 (2013).
7. Zhao, J., Gómez-Pérez, J. M., Belhajjame, K., *et al. Why workflows break—Understanding and combating decay in Taverna workflows* in *2012 IEEE 8th International Conference on E-Science* (2012), 1–9.
8. *Zip Format* `https://www.loc.gov/preservation/digital/formats/fdd/fdd000354.shtml`. Accessed: 1990-49-30.
9. Hucka, M., Finney, A., Sauro, H. M., *et al.* The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* **19,** 524–531 (2003).

10. Waltemath, D., Adams, R., Bergmann, F. T., *et al.* Reproducible computational biology experiments with SED-ML - The Simulation Experiment Description Markup Language. *BMC Syst. Biol.* **5,** 1–10 (2011).

11. Bergmann, F. T., Adams, R., Moodie, S., *et al.* COMBINE archive and OMEX format: one file to share all information to reproduce a modeling project. *BMC Bioinform.* **15,** 1–9 (2014).

12. Galdzicki, M., Rodriguez, C., Chandran, D., Sauro, H. M. & Gennari, J. H. Standard Biological Parts Knowledgebase. *PLoS ONE* **6,** e17005 (2011).

13. Galdzicki, M., Clancy, K. P., Oberortner, E., *et al.* The Synthetic Biology Open Language (SBOL) provides a community standard for communicating designs in synthetic biology. *Nat. Biotech.* **32,** 545–550 (2014).

14. Lloyd, C. M., Halstead, M. D. & Nielsen, P. F. CellML: its future, present and past. *Progress in biophysics and molecular biology* **85,** 433–450 (2004).

15. Madsen, C., Moreno, A. G., Umesh, P., *et al.* Synthetic Biology Open Language (SBOL) Version 2.3. *Journal of integrative bioinformatics* (2018).

16. Courtot, M., Juty, N., Knüpfer, C., *et al.* Controlled vocabularies and semantics in systems biology. *Molecular systems biology* **7,** 543 (2011).

17. Scharm, M., Waltemath, D., Mendes, P. & Wolkenhauer, O. COMODI: an ontology to characterise differences in versions of computational models in biology. *Journal of biomedical semantics* **7,** 46 (2016).

18. Dada, J. O., Spasić, I., Paton, N. W. & Mendes, P. SBRML: a markup language for associating systems biology data with models. *Bioinformatics* **26,** 932–938 (2010).

19. Le Novere, N., Hucka, M., Mi, H., *et al.* The systems biology graphical notation. *Nature biotechnology* **27,** 735 (2009).

20. Quinn, J. Y., Cox III, R. S., Adler, A., *et al.* SBOL visual: a graphical language for genetic designs. *PLoS biology* **13,** e1002310 (2015).

21. Gleeson, P., Crook, S., Cannon, R. C., *et al.* NeuroML: a language for describing data driven models of neurons and networks with a high degree of biological detail. *PLoS computational biology* **6,** e1000815 (2010).

22. Swat, M., Moodie, S., Wimalaratne, S., *et al.* Pharmacometrics Markup Language (PharmML): opening new perspectives for model exchange in drug development. *CPT: pharmacometrics & systems pharmacology* **4,** 316–319 (2015).

23. Smith, B., Ashburner, M., Rosse, C., *et al.* The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology* **25,** 1251 (2007).

24. Funahashi, A., Matsuoka, Y., Jouraku, A., *et al.* CellDesigner 3.5: A Versatile Modeling Tool for Biochemical Networks. *Proc. IEEE* **96,** 1254–1265 (2008).

25. Hoops, S., Sahle, S., Gauges, R., *et al.* COPASI—a COmplex PAthway SImulator. *Bioinformatics* **22,** 3067–3074 (2006).

26. Myers, C. J., Barker, N., Jones, K., *et al.* iBioSim: a tool for the analysis and design of genetic circuits. *Bioinformatics* **25,** 2848–2849 (2009).

27. Sauro, H. M. & Fell, D. *Jarnac: a system for interactive metabolic analysis* in *Animating the Cellular Map: Proceedings of the 9th International Meeting on BioThermoKinetics* (2000), 221–228.

28. Olivier, B. G. & Snoep, J. L. Web-based kinetic modelling using JWS Online. *Bioinformatics* **20,** 2143–2144 (2004).

29. Garny, A. & Hunter, P. J. OpenCOR: a modular and interoperable approach to computational biology. *Frontiers in physiology* **6,** 26 (2015).

30. Sauro, H. M. JDesigner: A simple biochemical network designer. *Available via the World Wide Web at http://members.tripod.co.uk/sauro/biotech.htm* (2001).

31. Olivier, B. G., Rohwer, J. M. & Hofmeyr, J.-H. S. Modelling cellular systems with PySCeS. *Bioinformatics* **21,** 560–561 (2005).

32. Bergmann, F. T. & Sauro, H. M. *SBW-a modular framework for systems biology* in *Proceedings of the 38th conference on Winter simulation* (2006), 1637–1645.

33. Choi, K., Medley, J. K., König, M., *et al.* Tellurium: An extensible python-based modeling environment for systems and synthetic biology. *Biosystems* **171,** 74–79 (2018).

34. Chandran, D., Bergmann, F. T. & Sauro, H. M. TinkerCell: modular CAD tool for synthetic biology. *Journal of biological engineering* **3,** 19 (2009).

35. Zhang, M., McLaughlin, J. A., Wipat, A. & Myers, C. J. SBOLDesigner 2: an intuitive tool for structural genetic design. *ACS synthetic biology* **6,** 1150–1160 (2017).

36. McLaughlin, J. A., Myers, C. J., Zundel, Z., *et al.* SynBioHub: A Standards-Enabled Design Repository for Synthetic Biology. *ACS synthetic biology* **7,** 682–688 (2018).

37. Keller, S., Kneissl, J., Grabher-Meier, V., *et al.* Evaluation of epidermal growth factor receptor signaling effects in gastric cancer cell lines by detailed motility-focused phenotypic characterization linked with molecular analysis. *BMC cancer* **17,** 845 (2017).

38. Heavner, B. D. & Price, N. D. Comparative analysis of yeast metabolic network models highlights progress, opportunities for metabolic reconstruction. *PLoS computational biology* **11,** e1004530 (2015).

39. Samal, S. S., Naldi, A., Grigoriev, D., *et al.* Geometric analysis of pathways dynamics: Application to versatility of TGF-β receptors. *Biosystems* **149,** 3–14 (2016).

40. Cooper, J., Spiteri, R. J. & Mirams, G. R. Cellular cardiac electrophysiology modeling with Chaste and CellML. *Frontiers in physiology* **5,** 511 (2015).

41. Cooper, J., Scharm, M. & Mirams, G. R. The cardiac electrophysiology web lab. *Biophysical journal* **110,** 292–300 (2016).

42. Beattie, K. A., Hill, A. P., Bardenet, R., *et al.* Sinusoidal voltage protocols for rapid characterisation of ion channel kinetics. *The Journal of physiology* **596,** 1813–1828 (2018).

43. Golding, N., August, T. A., Lucas, T. C., *et al.* The zoon R package for reproducible and shareable species distribution modelling. *Methods in Ecology and Evolution* **9,** 260–268 (2018).

44. Yang, P.-C., Purawat, S., Ieong, P. U., *et al.* A demonstration of modularity, reuse, reproducibility, portability and scalability for modeling and simulation of cardiac electrophysiology using Kepler Workflows. *PLOS Computational Biology* **15,** e1006856 (2019).

45. Gille, C., Bölling, C., Hoppe, A., *et al.* HepatoNet1: a comprehensive metabolic reconstruction of the human hepatocyte for the analysis of liver physiology. *Molecular systems biology* **6** (2010).

46. Mardinoglu, A., Agren, R., Kampf, C., *et al.* Genome-scale metabolic modelling of hepatocytes reveals serine deficiency in patients with non-alcoholic fatty liver disease. *Nature communications* **5,** 3083 (2014).

47. Agren, R., Bordel, S., Mardinoglu, A., *et al.* Reconstruction of genome-scale active metabolic networks for 69 human cell types and 16 cancer types using INIT. *PLoS computational biology* **8,** e1002518 (2012).

48. Maldonado, E. M., Fisher, C. P., Mazzatti, D. J., *et al.* Multi-scale, whole-system models of liver metabolic adaptation to fat and sugar in non-alcoholic fatty liver disease. *NPJ systems biology and applications* **4,** 33 (2018).

49. Krauss, M., Schaller, S., Borchers, S., *et al.* Integrating cellular metabolism into a multiscale whole-body model. *PLoS computational biology* **8,** e1002750 (2012).

50. Blais, E. M., Rawls, K. D., Dougherty, B. V., *et al.* Reconciled rat and human metabolic networks for comparative toxicogenomics and biomarker predictions. *Nature communications* **8,** 14250 (2017).

51. Szigeti, B., Gleeson, P., Vella, M., *et al.* OpenWorm: an open-science approach to modeling Caenorhabditis elegans. *Frontiers in computational neuroscience* **8,** 137 (2014).

52. Abbasi, N. A., Lafci, D. & Akan, O. B. Controlled information transfer through an in vivo nervous system. *Scientific reports* **8,** 2298 (2018).

53. Gleeson, P., Lung, D., Grosu, R., Hasani, R. & Larson, S. D. c302: a multiscale framework for modelling the nervous system of Caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences* **373,** 20170379 (2018).

54. Choi, K., Smith, L. P., Medley, J. K. & Sauro, H. M. phraSED-ML: A paraphrased, human-readable adaptation of SED-ML. *Journal of bioinformatics and computational biology* **14,** 1650035 (2016).

55. Ewald, R. & Uhrmacher, A. M. SESSL: A domain-specific language for simulation experiments. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* **24,** 11 (2014).

56. Leipzig, J. A review of bioinformatic pipeline frameworks. *Briefings in bioinformatics* **18,** 530–536 (2017).

57. Goecks, J., Nekrutenko, A. & Taylor, J. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome biology* **11,** R86 (2010).

58. Ludäscher, B., Altintas, I., Berkley, C., *et al.* Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience* **18,** 1039–1065 (2006).

59. Wolstencroft, K., Haines, R., Fellows, D., *et al.* The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic acids research* **41,** W557–W561 (2013).

60. Amstutz, P., Crusoe, M. R., Tijanić, N., *et al.* Common workflow language, v1. 0 (2016).

61. Goble, C. A., Bhagat, J., Aleksejevs, S., *et al.* myExperiment: a repository and social network for the sharing of bioinformatics workflows. *Nucleic acids research* **38,** W677–W682 (2010).

62. Vivian, J., Rao, A. A., Nothaft, F. A., *et al.* Toil enables reproducible, open source, big biomedical data analyses. *Nature biotechnology* **35,** 314 (2017).

63. Medley, J. K., Choi, K., König, M., *et al.* Tellurium notebooks—An environment for reproducible dynamical modeling in systems biology. *PLOS Computational Biology* **14,** 1–24 (2018).

64. Smith, L. P., Bergmann, F. T., Chandran, D. & Sauro, H. M. Antimony: a modular model definition language. *Bioinformatics* **25,** 2452–2454 (2009).

65. Consortium, G. O. The gene ontology project in 2008. *Nucleic acids research* **36,** D440–D444 (2007).

66. Neal, M. L., König, M., Nickerson, D., *et al.* Harmonizing semantic annotations for computational models in biology. *Briefings in bioinformatics* **20,** 540–550 (2018).

67. Missier, P., Belhajjame, K. & Cheney, J. *The W3C PROV family of specifications for modelling provenance metadata* in *Proceedings of the 16th International Conference on Extending Database Technology* (2013), 773–776.

68. Brush, M. H., Shefchek, K. & Haendel, M. *SEPIO: A Semantic Model for the Integration and Analysis of Scientific Evidence.* in *ICBO/BioCreative* (2016).

69. Lieven, C., Beber, M. E., Olivier, B. G., *et al.* Memote: A community-driven effort towards a standardized genome-scale metabolic model test suite. *BioRxiv,* 350991 (2018).

70. Gerkin, R. & Omar, C. NeuroUnit: Validation tests for neuroscience models. *Front Neuroinform* (2013).

71. Omar, C., Aldrich, J. & Gerkin, R. C. *Collaborative infrastructure for test-driven scientific model validation* in *Companion Proceedings of the 36th International Conference on Software Engineering* (2014), 524–527.