Toward standards for tomorrow's whole-cell models

Dagmar Waltemath[*,+], Jonathan R. Karr[+], Frank T. Bergmann, Vijayalakshmi Chelliah, Michael Hucka, Marcus Krantz, Wolfram Liebermeister, Pedro Mendes, Chris J. Myers, *Senior Member, IEEE,* Pinar Pir, Begum Alaybeyoglu, Naveen K Aranganathan, Kambiz Baghalian, Arne T. Bittig, Paulo E. Pinto Burke, Matteo Cantarelli, Yin Hoon Chew, Rafael S. Costa, Joseph Cursons, Tobias Czauderna, Harold F. Gómez, Jens Hahn, Tuure Hameri, Denis Kazakiewicz, Ilya Kiselev, Vincent Knight-Schrijver, Christian Knüpfer, Matthias König, Daewon Lee, Audald Lloret-Villas, Nikita Mandrik, J. Kyle Medley, Bertrand Moreau, Hojjat Naderi-Meshkin, Sucheendra K. Palaniappan, Daniel Priego-Espinosa, Martin Scharm, Mahesh Sharma, Kieran Smallbone, Natalie Stanford, Je-Hoon Song, Tom Theile, Milenko Tokic, Namrata Tomar, Jannis Uhlendorf, Thawfeek M Varusai, Florian Wendland, Markus Wolfien, James T. Yurkovich, Yan Zhu, Argyris Zardilis, Anna Zhukova, and Falk Schreiber[+]

Whole-cell modeling is a promising tool for biological research, bioengineering, and medicine. However, substantial work remains to create complete, accurate, and reproducible models. Among the advances needed are a strong theoretical understanding of multi-algorithm modeling, standardized modeling languages,

Manuscript received XXX XX, 2015; revised XXX XX, 201X; accepted XXX XX, 201X. Date of publication XXX XX, 201X; date of current version XXX XX, 201X. This work was supported in part by the Volkswagen Foundation (Grant to D. W. and F. S.) and the James S. McDonnell Foundation (Postdoctoral Fellowship Award in Studying Complex Systems to J. R. K.). *Asterisk indicates corresponding author: Dagmar Waltemath dagmar.waltemath@uni-rostock.de. Plus indicates these authors contributed equally to this work.*

[*]D. Waltemath, A. T. Bittig, M. Scharm, T. Theile, F. Wendland and M. Wolfien are with the Institute of Computer Science, University of Rostock, 18051 Rostock, Germany.

J. R. Karr is with the Department of Genetics & Genomic Sciences, Icahn School of Medicine at Mount Sinai, New York, NY 10029, USA.

C. J. Myers is with the Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, Utah 84112, USA.

M. Hucka is with the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA.

F. T. Bergmann is with BioQuant, University of Heidelberg, 69120 Heidelberg, Germany.

N. K. Aranganathan, A. Lloret-Villas and V. Chelliah are with the European Bioinformatics Institute (EMBL-EBI), European Molecular Biology Laboratory, Cambridge CB10 1SD, UK.

J. Hahn, M. Krantz and J. Uhlendorf are with the Department of Biology, Humboldt University of Berlin, 10115 Berlin, Germany.

W. Liebermeister and M. König are with the Institute of Biochemistry, University Medicine Charité Berlin, 10117 Berlin, Germany.

P. Mendes is with the Manchester Institute of Biotechnology and the School of Computer Science, University of Manchester, Manchester M1 7DN, UK and also with the Center for Quantitative Medicine and the Department of Cell Biology, University of Connecticut Health Center, Farmington, CT 06030, USA.

P. Pir and V. Knight-Schrijver are with the Babraham Institute, Cambridge CB22 3AT, UK.

B. Alaybeyoglu is with the Department of Chemical Engineering, Boğaziçi University, Bebek 34342, Turkey.

K. Baghalian is with the Department of Plant Sciences, University of Oxford, South Parks Road, Oxford, UK.

P. E. Pinto Burke is with the Institute of Science and Technology, Federal University of São Paulo, Brazil.

Y. H. Chew is with the Centre for Synthetic and Systems Biology, University of Edinburgh, Edinburgh EH9 3BF, UK.

M. Cantarelli is with OpenWorm.

R. S. Costa is with the Centre of Intelligent Systems-IDMEC, Instituto Superior Técnico, University of Lisbon, 1049-001 Lisboa, Portugal.

J. Cursons is with the Department of Biomedical Engineering, School of Engineering, University of Melbourne, Parkville, VIC 3010, Australia.

T. Czauderna is with the Faculty of Information Technology, Monash University, Clayton, VIC 3800, Australia.

H. F. Gómez is with the Department of Biosystems Science and Engineering, ETH Zürich, Basel, Switzerland.

T. Hameri and M. Tokic are with the Laboratory of Computational Systems Biotechnology (LCSB), Swiss Federal Institute of Technology (EPFL), CH-1015 Lausanne, Switzerland. M. Tokic is also with the Swiss Institute of Bioinformatics (SIB), CH-1015 Switzerland.

D. Kazakiewicz is with the Center for Statistics, Universiteit Hasselt, Hasselt BE3500, Belgium, and also with the Center for Innovative Research, Medical University of Białystok, Białystok 15-089, Poland.

I. Kiselev is with the Design Technological Institute of Digital Techniques, Siberian Branch of the Russian Academy of Sciences, Novosibirsk 630090, Russia.

C. Knüpfer is with the Institut für Informatik, University of Jena, 07743 Jena, Germany.

D. Lee and J.-H. Song are with the Department of Bio and Brain Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Republic of Korea.

N. Mandrik is with the Sobolev Institute of Mathematics, Siberian Branch of the Russian Academy of Sciences, Novosibirsk 630090, Russia.

J. K. Medley is with the Department of Bioengineering, University of Washington, Seattle, WA 98195, USA.

H. Naderi-Meshkin is with the Stem Cell and Regenerative Medicine Research Department, Iranian Academic Center for Education, Culture Research (ACECR), Khorasan Razavi Branch, Mashhad, Iran.

B. Moreau is with the CoSMo Company, Lyon, France.

S. K. Palaniappan is with the Rennes - Bretagne Atlantique Research Centre, Institute for Research in Computer Science and Automation, 35042 Rennes Cedex, France.

D. Priego-Espinosa is with the Instituto de Ciencias Físicas, Universidad Nacional Autónoma de México, México.

M. Sharma is with the Department of Pharmacoinformatics, National Institute of Pharmaceutical Education and Research, Punjab 160062, India.

K. Smallbone and N. Stanford are with the Manchester Centre for Integrative Systems Biology, University of Manchester, Manchester M1 7DN, UK.

N. Tomar is with the Department of Dermatology, University Medicine, Friedrich-Alexander University of Erlangen-Nürnberg, Erlangen, Germany.

T. M. Varusai is with the Department of Systems Biology Ireland, University College Dublin, Belfield, Dublin 4, Ireland.

J. T. Yurkovich is with the Department of Bioengineering, University of California, San Diego, La Jolla, CA 92093, USA.

Y. Zhu is with Monash Institute of Pharmaceutical Sciences, Monash University, Parkville, VIC 3052, Australia.

A. Zardilis is with the Centre for Synthetic and Systems Biology, University of Edinburgh, UK.

A. Zhukova is with the Institut de Biochimie et Génétique Cellulaires, National Center for Scientific Research, and also with the University of Bordeaux, France, 33077 Bordeaux Cedex, France.

F. Schreiber is with the Faculty of Information Technology, Monash University, Clayton, VIC 3800, Australia and also with the Institute of Computer Science, Martin Luther University Halle-Wittenberg, 06108 Halle, Germany.

Digital Object Identifier 10.1109/TBME.XXXX.XXXXXXX

and an efficient general-purpose simulator. We organized the 2015 Whole-Cell Modeling Summer School to teach whole-cell modeling, as well as to evaluate the need for new modeling standards and tools by encoding a recently published whole-cell model in SBML. We describe several standards extensions, software tools, and databases which are needed to facilitate reproducible whole-cell modeling, including a graphical model editor, a multi-algorithm simulator, and several SBGN extensions. Together these new standard extensions and software tools could accelerate whole-cell modeling.

***Index Terms*—Whole-cell modeling, Systems biology, Computational biology, Simulation, Standards, Education**

I. Introduction

OVER the past twenty years, computational modeling has become an essential and powerful tool for biological research, bioengineering, and medicine to analyze high-throughput molecular measurements and understand the molecular details of complex biological systems. Computational modeling has been used to identify new metabolic genes [1], to engineer metabolic pathways in bacteria [2], and to identify potential new antimicrobial drug targets [3]. Computational models also have the potential to enable bioengineers to design new microorganisms for industrial applications such as chemical synthesis, biofuel production, and waste decontamination, as well as to enable clinicians to tailor therapy to individual patients. Realizing this potential requires more comprehensive and accurate computational models that are capable of predicting cellular behavior from genotype. Realizing this potential also requires improved simulation tools, as well as standardized tools for storing and exchanging models, simulation experiments, and visualizations [4–9].

Recently, researchers at Stanford University developed the first whole-cell model of the gram-positive bacterium *Mycoplasma genitalium* [10]. The model represents the life cycle of a single Mycoplasma cell including the copy number dynamics of each metabolite, RNA, and protein species and accounts for every known gene function. The model is composed of 28 submodels, each of which is implemented using different mathematical formalisms including *ordinary differential equations* (ODEs), *flux balance analysis* (FBA), and *Boolean rules* (BRs), and trained using different experimental data. The model is implemented in MATLAB, is available open-source under the MIT license [11], and is extensively documented. This has enabled other researchers to use the model to conduct *in silico* experiments [12–14].

Despite extensive documentation, this whole-cell model software is difficult to use. The software is large, complex, and time-consuming to learn. The software also cannot easily be reused to simulate other models because many of the model details are intertwined with the software. In addition, many academic researchers and many software developers cannot use the model software because MATLAB is proprietary and expensive. Compared to other languages with larger development communities such as Python, MATLAB has also few packages and development tools. In particular, MATLAB has limited support for object-oriented programming. Furthermore, the model software is not optimally efficient because MATLAB is just-in-time compiled and dynamically typed.

Covert and colleagues developed the WholeCellKB [15], WholeCellSimDB [16], and WholeCellViz [17] software tools to provide user-friendly interfaces to their modeling software. However, significant domain expertise is still required to use their modeling software, particularly to construct new models. Alternative approaches are needed to enable more researchers to develop and simulate their own whole-cell models. Software-independent standards and other open-source software tools have the potential to enable researchers to develop such models more quickly, to explore them more deeply, and to evaluate them more rigorously. Furthermore, standards would make whole-cell models more reusable and comparable, as well as more searchable and retrievable through model repositories such as BioModels [18, 19].

Several software-agnostic systems biology standards have already been developed by the *COmputational Modeling in BIology NEtwork* (COMBINE) [20], including the *Systems Biology Markup Language* (SBML) [21], the *Cell Markup Language* (CellML) [22], the *Simulation Experiment Description Markup Language* (SED-ML) [23], and the *Systems Biology Graphical Notation* (SBGN) [24] (see Table I). SBML and CellML are languages for representing mathematical models. CellML focuses on modularly encoding mathematical models, while SBML focuses on describing biological processes. Both support several modeling formalisms including ODEs and FBA. SED-ML is a language for describing computational experiments, including the simulation algorithm and parameter values. SED-ML enables scientists to reproduce simulations. SBGN is a visual notation for describing biological processes. To date, none of these open standards have been used with models as complex as the *M. genitalium* model.

We organized the 2015 Whole-Cell Modeling Summer School to train students in whole-cell modeling, as well as to evaluate the need for new standards and tools for whole-cell modeling. The majority of the school focused on encoding the *M. genitalium* model using SBML, visualizing it with SBGN, and simulating it with SED-ML. This was both a learning exercise, as well as a tool for evaluating standard representations capabilities for encoding whole-cell models. The ultimate goal of the school was to encode an open-source whole-cell model in SBML, visualize the model using SBGN, simulate the model using open-source software, and use the model to conduct *in silico* experiments encoded in SED-ML. We chose to focus the course on SBML because there was insufficient time to evaluate both SBML and CellML.

Here, we describe the summer school, outline our progress encoding the *M. genitalium* model using standards, and describe several standard extensions and software tools that we believe are needed to support efficient whole-cell modeling.

II. The 2015 Whole-Cell Modeling Summer School

We organized the summer school to teach students how to build and encode models using COMBINE standards by encoding the *M. genitalium* model using software-independent, standard representation formats.

Table I
STANDARDS AND INFRASTRUCTURE USED DURING THE SUMMER SCHOOL.

Acronym	Name	Description
COMBINE	COmputational Modeling in BIology NEtwork	Community dedicated to developing standards and associated software tools
SBGN	Systems Biology Graphical Notation	Standard for describing biochemical pathway diagrams
SBML Level 3 Core	Systems Biology Markup Language	Standard for describing dynamical models in terms of biochemical processes
SBML Comp	SBML Package: Hierarchical Model Composition	Definition of how a model is composed from other models [25]
SBML FBC	SBML Package: Flux Balance Constraints	Definition of constraint based models [26]
SBML Multi	SBML Package: Multistate and Multicomponent Species	Representation of entity pools with multiple states and composed of multiple components [27]
SBML Arrays	SBML Package: Arrays	Support for expressing arrays of things [27]
SED-ML	Simulation Experiment Description Markup Language	Standard for describing computational experiments

A. *Organization*

The Whole-Cell Modeling Summer School was held March 9-13, 2015 at the University of Rostock, Germany. The school was organized by Dagmar Waltemath and Falk Schreiber and supported by the Volkswagen Foundation. The school included 43 students, nine instructors, and two organizers.

The school began with two introductory lectures on modeling and modeling standards. Jonathan Karr from the Icahn School of Medicine at Mount Sinai, USA, presented an overview of whole-cell and multi-algorithm modeling. Michael Hucka from the California Institute of Technology, USA, presented an overview of the SBML, SED-ML, and SBGN standards; open-source software tools that support these standards; and the COMBINE initiative. We also organized three discussions on model composition, particle-based state representation, and stochastic modeling.

The majority of the school was devoted to hands-on learning about whole-cell modeling and the COMBINE standards by encoding the *M. genitalium* model in SBML, creating visualizations using SBGN, and defining simulations using SED-ML. The 28 sub-models were divided among nine groups of four to five students and one instructor. Each day concluded with brief progress reports from each group to exchange ideas and facilitate discussion. We also organized a poster session and several social activities to encourage students to network.

B. *Educational outcome*

Most students reported gaining knowledge of whole-cell modeling, increased appreciation for reproducibility, and increased understanding of the SBML, SED-ML, and SBGN standards. Many students also reported learning about open-source software tools relevant to their own research.

In addition, many of the students reported that the school expanded their network. The school introduced several students to other workshops and job opportunities.

C. *Lessons learned for organizing research-based schools*

We learned several valuable lessons about how to best organize a research-based school. First, we learned that research-based schools should have clear background knowledge expectations and learning objectives and have well-planned learning exercises. This helps students make informed decisions about whether to participate in the school, know how to prepare for the school, and learn efficiently. Second, we found that students greatly enjoy learning through open research problems rather than through prescribed training exercises. This challenges students and engages them in research. This also helps students build practical skills that complement their undergraduate training. Third, we found that open-ended project-based schools require a high teacher-to-student ratio, a flexible schedule, and multidisciplinary project teams. A high teacher to student ratio allows students to get feedback and iterate through potential solutions quickly. A flexible schedule enables impromptu lectures and discussions. Multidisciplinary teams enable students to work through difficult problems by drawing on perspectives from multiple fields.

III. TOWARD AN SBML-ENCODED WHOLE-CELL MODEL

The second goal of the school was to encode the *M. genitalium* whole-cell model in SBML. To achieve this goal, most of the course was devoted to active learning sessions in which students were challenged to encode submodels of the *M. genitalium* in SBML, integrate submodels into a single model, and simulate models using SED-ML. During these sessions, the students and instructors were divided into nine groups. Eight of the groups were tasked with encoding one or more submodels. The ninth group was tasked with developing a standards-compliant scheme to integrate the submodels into a single model. In addition, three instructors helped all of the groups annotate and visualize their submodels and one instructor helped all of the groups encode their submodels in SBML. Table SI in the supplemental material lists the nine groups and all of the students and instructors.

A. *Submodel encoding*

The eight submodel encoding groups pursued various strategies to encode submodels in SBML. Several of the groups encoded submodels by first reading the submodel documentation, then drawing pathway diagrams using software tools such as CellDesigner [28] and VANTED [29], and finally writing scripts to generate SBML models from their diagrams using libSBML [30]. Other groups used modeling software tools such as Antimony [31], BioUML [32], COBRApy [33], COPASI [34], iBioSim [35], and libRoadRunner [36] to encode submodels based on their documentation. A few of the groups encoded submodels by converting the MATLAB code to SBML. These groups then generated SBGN diagrams from their SBML to better understand their submodels.

The groups encountered several challenges to encoding the submodels in SBML. First, understanding the submodels was time-consuming because the documentation only summarizes the submodels, the connection between the submodels and the associated pathway/genome database is unclear, and many of the submodels details are implemented directly in the MATLAB code. Fortunately, one of the principal authors of the *M. genitalium* model participated in the school.

A second challenge to encoding the submodels was to encode serially-executed MATLAB submodels in SBML, because SBML does not explicitly represent sequential operations. Most of the groups decided to tackle this problem by formalizing MATLAB submodels as discrete stochastic models and simulating them using the Gillespie stochastic simulation algorithm [37]. One drawback of this approach is that it required assigning kinetics that are not present in the original MATLAB submodels due to insufficient kinetic data.

A third challenge to encoding the submodels was to encode the randomized algorithms used by the MATLAB submodels in SBML. For example, the MATLAB translation submodel includes a randomized algorithm that assigns amino acids to individual polypeptides. This algorithm is not equivalent to the Gillespie algorithm and cannot easily be encoded in SBML because plain SBML does not support random number generation. Most of the groups also solved this problem by formalizing submodels as stochastic models.

To encode many of the submodels in SBML, the groups also had to either enumerate the particle-based state representation used by the MATLAB submodels, or approximate the MATLAB submodels. The translation group chose to approximate their submodel by eliminating the internal dynamics of the polymerization of each polypeptide. Consequently, their submodel does not track the progress of individual ribosomes or account for base-specific translation rates. The replication, replication initiation, transcription, and transcriptional regulation groups, chose to enumerate the chromosome representation used by the MATLAB model by creating Boolean indicator variables to represent the existence and protein-binding status of each base. This enumerated representation requires millions of variables. Consequently, the corresponding SBML files are computationally expensive to parse and simulate. Enumerating the rules that govern the joint values of the enumerated variables, such as the rules that represent the steric effects of DNA-bound proteins by preventing proteins from binding neighboring bases, is also impractical. Furthermore, editing this enumerated representation is difficult because thousands of variables and rules must be edited rather than a small number of variable and rule patterns.

The lack of SBML simulator support for arrays was another challenge to encoding submodels in SBML. All of the groups overcame the lack of array support by enumerating individual array elements and all matrix algebra computations. This created verbose SBML files that are difficult to interpret, maintain, and edit. Enumerating the matrix algebra computations also increases the computational cost of simulation.

Combined, we found it difficult to encode most of the MATLAB submodels in SBML. As discussed below, future progress in whole-cell modeling would be facilitated by expanded support for the SBML Hierarchical Model Composition, Multistate and Multicomponent Species [38] and Arrays [39] packages.

B. Model integration

The integration group was responsible for assembling the submodels into a single model. This included devising a scheme for representing global state variables, defining the interfaces exposed by the submodels to the global state variables, and developing a method to manage concurrent writing of shared state variables by multiple submodels. The integration group defined the global state variables as the union of all state variables shared by at least two submodels rather than by explicitly defining a set of global state variables as done by the original MATLAB-based simulation. The advantages of this approach are that submodel developers are not also required to develop global state variables and that it minimizes the number of global state variables. The disadvantages of this approach are that the total set of variables is less transparent and that it requires users to learn all of the submodels and their naming conventions to analyze model simulations.

The integration group standardized the submodel interfaces by defining a variable naming convention. This convention makes it clear how multiple local submodel variables map onto the same global variable. The integration group used the same variable names as those used by the MATLAB implementation. Matrix and particle-based variables were enumerated by creating multiple variables differentiated by the suffixes.

The primary challenge faced by the integration group was managing concurrent editing of state variables by multiple submodels. The group explored several potential solutions. First, they explored sequentially simulating the submodels and updating the global state variables. This avoids needing to merge variable changes. However, under this approach, submodels are simulated with different variable values within each time step. Consequently, simulation predictions are sensitive to the submodel execution order.

The integration group also explored several more complex solutions that would enable submodels to be simulated with the same variable values within each time step. These strategies included reducing the submodel integration time step so that submodels do not request conflicting variable changes; dividing each of the shared state variables into multiple, independent sub-variables for each submodel, simulating the submodels, and merging the sub-variables to update global values; and using semaphores to manage concurrent variable editing whereby at each time step submodels request sets of atomic state changes and a controller decides which change sets are accepted. Each of these strategies has advantages and disadvantages. The first strategy is simple to understand and implement, but is computationally expensive. The second strategy is simple to implement and computationally efficient for independent variables, but is complex for coupled variables such as those that represent the chromosome protein

Table II
NEW AND EXPANDED TOOLS NEEDED TO FACILITATE WHOLE-CELL MODELING.

Type	Description
Database	Expanded molecular biological databases such as ChEBI
Software	Efficient, parallelized multi-algorithm simulator which supports the SBML Hierarchical Model Composition package
Software	Hybrid population/particle simulator which supports the SBML Multistate and Multicomponent Species package
Software	Graphical model editor which transparently build models from pathway/genome databases
Standard and software	SBGN standard and tool support for hybrid maps containing Process Description, Entity Relationship, and Activity Flow nodes

occupancy. The third strategy is complex, but is more general than the second strategy and more computationally efficient than the first. Ultimately, we concluded that a combination of these strategies will be needed to efficiently manage concurrent writing of different types of shared variables.

The integration group implemented their strategies using SBML and SED-ML. In particular, the group wrote Python scripts to create SBML and SED-ML files automatically from the variable usage information provided by the other teams. The first script created template SBML models for each process to ensure consistent interfaces. The second script created a SED-ML file to implement the coordinated execution of all the current processes and share the variables using the schemes described above. The third script created a hierarchical SBML model that connected the processes in one global top-level file, and it also included separate models to implement variable sharing between the processes. The SED-ML and hierarchical SBML approaches are two alternatives that we decided to explore in parallel. Which one is ultimately used depends on whether one feels the description of the execution of the parallel processes is part of the model or the simulation strategy.

The integration group tested their strategies using iBioSim because iBioSim is one of the few simulators that supports model composition. Another challenge to integrating the submodels was the lack of a multi-algorithm simulator. The integration group plans to overcome this limitation by adding support for multi-algorithm simulation to iBioSim.

C. Annotation, documentation, and visualization

The annotation, documentation and visualization group was responsible for annotating the model. The goal was to annotate every model entity with a cross reference to an external database such as ChEBI [40], as in the case of small molecules, and/or in terms of other model entities, as in the case of chemical reactions. The group wrote scripts to search molecular biology databases for every entity contained in the *M. genitalium* model. The main problem faced by the documentation group was that many model entities are not currently represented by any molecular biology database. This shortcoming can easily be overcome by proposing new ontology terms to the databases. This problem highlights the need to expand the molecular biology databases to aggregate data on more biological molecules and interactions.

The documentation group also helped the other groups visualize submodels by providing advice on SBGN and diagramming tools such as VANTED [29]. The main visualization problem encountered by the group was that whole-cell models require large diagrams that must be manually arranged to produce intuitive visualizations.

D. Progress

We produced preliminary SBML and SBGN versions of most of the *M. genitalium* submodels and finished SBML and SBGN versions for the cytokinesis submodel. Significant work remains to finish encoding the model. We must finish encoding the submodels, expand software tools such as iBioSim to support multi-algorithm modeling, rigorously test the entire SBML model by reproducing all of the MATLAB model tests, and thoroughly diagram and document the SBML model.

E. Future steps

We hope to finish encoding the *M. genitalium* submodels in SBML and integrating the SBML-encoded submodels into a single model that can be simulated by open-source software tools such as BioUML, COPASI, iBioSim, and libRoadRunner. Several students and instructors have continued to work and meet online. Many also plan to participate in a second meeting in October, 2015, that will be held at the University of Utah, USA, immediately prior to the 2015 COMBINE Forum (http://co.mbine.org/events/COMBINE_2015).

Going forward, we hope to publish SBML-encoded versions of each of the *M. genitalium* submodels to BioModels, along with SED-ML tests, SBGN diagrams, and textual documentations. This would make the submodels searchable, retrievable, and reusable by other scientists. We believe this would be a valuable community resource. It would demonstrate how to build a whole-cell model and enable other researchers to build upon the *M. genitalium* submodels.

IV. TOWARD SBML-, SED-ML-, AND SBGN-BASED STANDARDS FOR WHOLE-CELL MODELING

The school was the first attempt to encode a whole-cell model using open standards. Thus we were not surprised to learn several limitations with currently-available formats. More importantly, the school generated ideas for how our existing standards and tools can be expanded to better support large, heterogeneous models.

A. Standard extensions

Several enhancements to simulation software tools and databases are needed to facilitate whole-cell modeling (Table II). First, more SBML simulators must support multi-algorithm simulations. This will require research to determine the best way to integrate heterogeneous submodels, including rigorously evaluating the schemes proposed by the integration group. Significant effort will also be needed to develop an efficient, parallelized, multi-algorithm simulator.

Second, more SBML simulators must implement the SBML Multistate and Multicomponent Species package to support hybrid population/particle-based state representation such as that used by BioNetGen [41, 42] and NFSim [43]. This would enable more succinct model descriptions, making models much easier to understand, edit, and expand. For example, translation could be described using a single reaction pattern and parameterized by arrays of mRNA-specific translation initiation rates and codon-specific elongation rates rather than by enumerating each reaction. By separating mathematical descriptions from quantitative parameter values, reaction patterns would also make the connection between dynamical models and the experimental data used to inform their parameters more transparent. Implementing the SBML Multistate and Multicomponent Species package would also enable modelers to efficiently simulate models with large, combinatorial state spaces.

New user-friendly graphical editors must also be developed to enable researchers to easily build SBML files with these new features. These graphical editors must also allow researchers to transparently map model parameters onto experimental data organized in pathway/genome databases.

In addition, molecular biology databases such as ChEBI must be expanded to enable researchers to concretely define whole-cell models in terms of external entities.

SBGN must also be expanded in several ways. SBGN should support hybrid diagrams which contain elements from Process Description, Entity Relationship, and Activity Flow. It also needs to support generic reactions to reduce the size of a diagram and the amount of repetitive reaction chains with simple modifications of reactions partners, such as fatty acid synthesis. The SBGN viewers must also be expanded to provide advanced layout algorithms suitable for large visualizations and automatically display diagrams at multiple levels of granularity using contextual zooming and model reduction.

Together, these software, database, and SBGN expansions would enable more researchers to more easily build, manage, simulate, and reproduce whole-cell models and simulations. These new standards and tools would also enable researchers to build more comprehensive and more accurate models. Ultimately, this would enable whole-cell modeling to support bioengineering and personalized medicine.

B. *The whole-cell modeling pipeline*

We anticipate that such expanded standards and tools will enable a four step approach to whole-cell model-driven discovery (Fig. 1). First, researchers will assemble experimental data from numerous sources including databases such as SABIO-RK [44] and UniProt [45] into pathway/genome databases using software tools such as Pathway Tools [46] and WholeCellKB. Second, researchers will use pathway/genome databases and graphical modeling tools such as BioUML, COPASI, and iBioSim to build submodels and encode them using transparent languages such as SBML. Third, multi-algorithm simulators will be used to conduct *in silico* experiments. Lastly, software tools such as WholeCellSimDB and WholeCellViz will be used to discover new biology through exploring, visualizing, and analyzing in silico experiments.

V. Conclusion

The 2015 Whole-Cell Modeling Summer School provided 43 young scientists training in whole-cell and multi-algorithm modeling by encoding the *M. genitalium* whole-cell model in SBML. Additional courses are needed to provide students with deeper theoretical training in dynamical modeling, multi-algorithm modeling, model reduction, and parameter estimation, as well as practical training in model construction including data curation, model building, numerical optimization, model testing, and model analysis.

Significant strides were made toward implementing the *M. genitalium* whole-cell model using SBML. We developed preliminary SBML versions of all of the submodels of the *M. genitalium* model. We have continued to encode the *M. genitalium* model since the school. Ultimately, we hope to publish an SBML-encoded version of the model to BioModels.

In addition, the summer school generated clear goals for expanding the existing SBML software tools to support whole-cell modeling. The SBML simulators must be expanded to support all of the Hierarchical Model Composition, Multistate and Multicomponent Species, Arrays, and Flux Balance Constraints packages to efficiently simulate whole-cell models. Furthermore, the school unified modelers, software developers, and standards developers to develop standardized, open-source tools for whole-cell and other large models.

New parameter estimation, model testing, and visual analysis tools must also be developed to enable researchers to effectively use SBML-encoded whole-cell models for research. In addition, our molecular biology databases must be expanded to facilitate whole-cell model annotation. Furthermore, SBGN and the SBGN viewers must be expanded to support hybrid diagrams, advanced automatic graph layout, automatic graph reduction, and contextual zooming. CellML should also be rigorously evaluated as another potential whole-cell modeling standard.

In summary, we believe that whole-cell modeling has the potential to be an important tool for biological discovery, bioengineering, and medicine. Achieving this potential requires new simulation software for simulating whole-cell models. In turn, this requires expanding the whole-cell modeling field including training young researchers.

References

[1] J. L. Reed, T. R. Patel, K. H. Chen *et al.*, “Systems approach to refining genome annotation,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 103, no. 46, pp. 17 480–17 484, 2006.

[2] J. W. Lee, D. Na, J. M. Park *et al.*, “Systems metabolic engineering of microorganisms for natural and non-natural chemicals,” *Nat. Chem. Biol.*, vol. 8, no. 6, pp. 536–546, 2012.

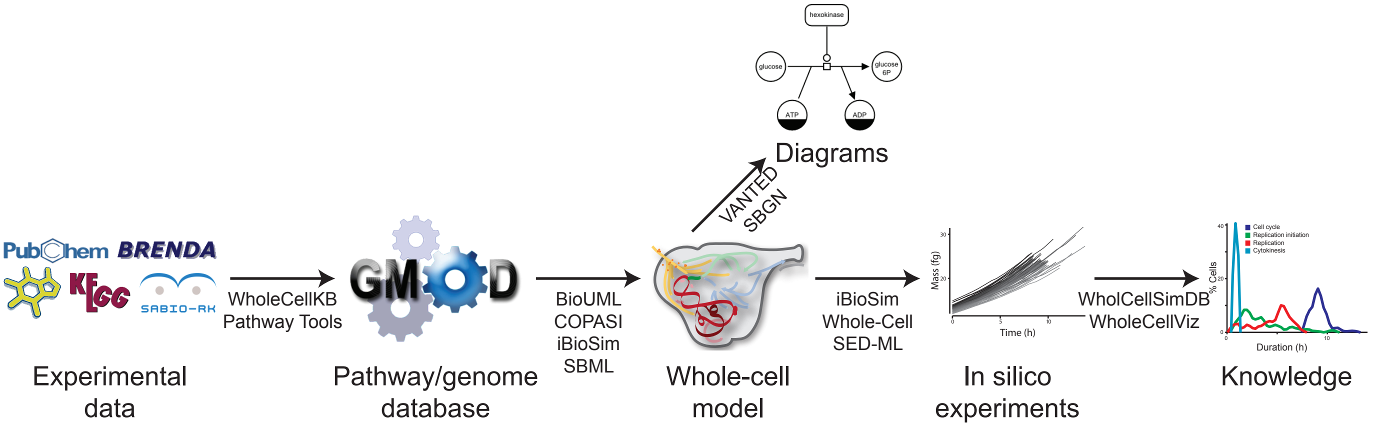

Figure 1. Whole-cell modeling pipeline. First, researchers will assemble experimental data into pathway/genome databases. Second, researchers will use pathway/genome databases to construct submodels. Third, researchers will use multi-algorithm simulators to conduct *in silico* experiments. Lastly, researchers will analyze in silico experiments to learn new biology.

[3] D. S. Lee, H. Burd, J. Liu *et al.*, "Comparative genome-scale metabolic reconstruction and flux balance analysis of multiple *Staphylococcus aureus* genomes identify novel antimicrobial drug targets," *J. Bacteriol.*, vol. 191, no. 12, pp. 4015–4024, 2009.

[4] D. N. Macklin, N. A. Ruggero, and M. W. Covert, "The future of whole-cell modeling," *Curr. Opin. Biotechnol.*, vol. 28, pp. 111–115, 2014.

[5] J. R. Karr, K. Takahashi, and A. Funahashi, "The principles of whole-cell modeling," *Curr. Opin. Microbiol.*, vol. (in press), 2015.

[6] J. R. Karr, A. H. Williams, J. D. Zucker *et al.*, "Summary of the DREAM8 Parameter Estimation Challenge: Toward parameter identification for whole-cell models," *PLoS Comput. Biol.*, vol. 11, no. 5, p. e1004096, 2015.

[7] M. Hucka, D. P. Nickerson, G. D. Bader *et al.*, "Promoting coordinated development of community-based information standards for modeling in biology: the COMBINE initiative," *Front. Bioeng. Biotechnol.*, vol. 3, 2015.

[8] E. Klipp, W. Liebermeister, A. Helbig *et al.*, "Systems biology standards—the community speaks," *Nat. Biotechnol.*, vol. 25, no. 4, pp. 390–391, 2007.

[9] F. Büchel, N. Rodriguez, N. Swainston *et al.*, "Path2Models: large-scale generation of computational models from biochemical pathway maps," *BMC Syst. Biol.*, vol. 7, no. 1, p. 116, 2013.

[10] J. R. Karr, J. C. Sanghvi, D. N. Macklin *et al.*, "A whole-cell computational model predicts phenotype from genotype," *Cell*, vol. 150, no. 2, pp. 389–401, 2012.

[11] ——. Whole-Cell model version 1.1. [Online]. Available: https://github.com/CovertLab/WholeCell/releases/tag/v1.1

[12] J. C. Sanghvi, S. Regot, S. Carrasco *et al.*, "Accelerated discovery via a whole-cell model," *Nat. Methods*, vol. 10, no. 12, pp. 1192–1195, 2013.

[13] O. Purcell, B. Jain, J. R. Karr *et al.*, "Towards a whole-cell modeling approach for synthetic biology," *Chaos*, vol. 23, no. 2, p. 025112, 2013.

[14] D. Kazakiewicz, J. R. Karr, K. Langner, and D. Plewczynski, "A combined systems and structural modeling approach repositions antibiotics for mycoplasma genitalium," *Comput. Biol. Chem.*, vol. 58, no. XX–YY, 2015.

[15] J. R. Karr, J. C. Sanghvi, D. N. Macklin *et al.*, "WholeCellKB: model organism databases for comprehensive whole-cell models," *Nucleic Acids Res.*, vol. 41, no. Database issue, pp. D787–D792, 2013.

[16] J. R. Karr, N. C. Phillips, and M. W. Covert, "WholeCellSimDB: a hybrid relational/hdf database for whole-cell model predictions," *Database*, vol. 2014, no. pii, p. bau095, 2014.

[17] R. Lee, J. R. Karr, and M. W. Covert, "WholeCellViz: data visualization for whole-cell models," *BMC Bioinformatics*, vol. 14, p. 253, 2013.

[18] N. Juty, R. Ali, M. Glont *et al.*, "BioModels: Content, features, functionality, and use," *CPT Pharmacometrics Syst. Pharmacol.*, vol. 4, no. 2, pp. 1–14, 2015.

[19] V. Chelliah, N. Juty, I. Ajmera *et al.*, "BioModels: ten-year anniversary," *Nucleic Acids Res.*, vol. 43, no. D1, pp. D542–D548, 2015.

[20] N. Le Novère, M. Hucka, N. Anwar *et al.*, "Meeting report from the first meetings of the Computational Modeling in Biology Network (COMBINE)," *Stand. Genomic Sci.*, vol. 5, no. 2, p. 230, 2011.

[21] M. Hucka, A. Finney, H. M. Sauro *et al.*, "The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models," *Bioinformatics*, vol. 19, no. 4, pp. 524–531, 2003.

[22] W. J. Hedley, M. R. Nelson, D. P. Bullivant, and P. F. Nielson, "A short introduction to CellML," *Philos. Trans. R. Soc. Lond. A*, vol. 359, pp. 1073–1089, 2001.

[23] D. Waltemath, R. Adams, F. Bergmann *et al.*, "Reproducible computational biology experiments with SED-ML—the Simulation Experiment Description Markup Language," *BMC Syst. Biol.*, vol. 5, no. 1, p. 198, 2011.

[24] N. Le Novère, M. Hucka, H. Mi *et al.*, "The Systems Biology Graphical Notation," *Nat. Biotechnol.*, vol. 27,

pp. 735–741, 2009.

[25] L. P. Smith, M. Hucka, S. Hoops *et al.*, "Sbml level 3 package specification: Hierarchical model composition," 2013.

[26] B. G. Olivier and F. T. Bergmann, "Sbml level 3 package: Flux balance constraints ('fbc')," 2013.

[27] D. Waltemath, F. T. Bergmann, C. Chaouiya *et al.*, "Meeting report from the fourth meeting of the Computational Modeling in Biology Network (COMBINE)," *Stand. Genomic Sci.*, vol. 9, no. 3, 2014.

[28] A. Funahashi, Y. Matsuoka, A. Jouraku *et al.*, "CellDesigner 3.5: a versatile modeling tool for biochemical networks," *Proc. IEEE*, vol. 96, no. 8, pp. 1254–1265, 2008.

[29] H. Rohn, A. Junker, A. Hartmann *et al.*, "VANTED v2: a framework for systems biology applications," *BMC Syst. Biol.*, vol. 6, p. 139, 2012.

[30] B. J. Bornstein, S. M. Keating, A. Jouraku, and M. Hucka, "LibSBML: an API library for SBML," *Bioinformatics*, vol. 24, no. 6, pp. 880–881, 2008.

[31] L. P. Smith, F. T. Bergmann, D. Chandran, and H. M. Sauro, "Antimony: a modular model definition language," *Bioinformatics*, vol. 25, no. 18, pp. 2452–2454, 2009.

[32] F. Kolpakov, "BioUML: visual modeling, automated code generation and simulation of biological systems," *Proc. BGRS*, vol. 3, pp. 281–285, 2006.

[33] A. Ebrahim, J. A. Lerman, B. O. Palsson, and D. R. Hyduke, "COBRApy: constraints-based reconstruction and analysis for python," *BMC Syst. Biol.*, vol. 7, no. 1, p. 74, 2013.

[34] P. Mendes, S. Hoops, S. Sahle *et al.*, "Computational modeling of biochemical networks using COPASI," *Methods Mol. Biol.*, vol. 500, pp. 17–59, 2009.

[35] C. Madsen, C. J. Myers, T. Patterson *et al.*, "Design and test of genetic circuits using ibiosim," *IEEE Des. Test Comput.*, vol. 29, no. 3, 2012.

[36] E. T. Somogyi, J.-M. Bouteiller, J. A. Glazier *et al.*, "libRoadRunner: a high performance SBML simulation and analysis library," *Bioinformatics*, 2015. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/early/2015/07/13/bioinformatics.btv363.abstract

[37] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *J. Phys. Chem.*, vol. 81, no. 25, pp. 2340–2361, 1977. [Online]. Available: http://dx.doi.org/10.1021/j100540a008

[38] F. Zhang and M. Meier-Schellersheim, "SBML Level 3 Package Specification: Multistate/Multicomponent Species (Version 1, Release 0.1 Draft 369)," 2015, accessed: 2015-05-25. [Online]. Available: http://sbml.org/Documents/Specifications/SBML_Level_3/Packages/multi

[39] L. Watanabe, C. J. Myers, and L. P. Smith, "SBML Level 3 Package Specification: Arrays (Draft of April 6, 2015)," 2015, accessed: 2015-05-25. [Online]. Available: http://sbml.org/Documents/Specifications/SBML_Level_3/Packages/arrays

[40] J. Hastings, P. de Matos, A. Dekker *et al.*, "The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013," *Nucleic Acids Res.*, vol. 41, no. Database issue, pp. D456–D463, 2013.

[41] W. S. Hlavacek, J. R. Faeder, M. L. Blinov *et al.*, "Rules for modeling signal-transduction systems," *Sci. STKE*, vol. 2006, no. 344, p. re6, 2006.

[42] J. S. Hogg, L. A. Harris, L. J. Stover *et al.*, "Exact hybrid particle/population simulation of rule-based models of biochemical systems," *PLoS Comput. Biol.*, vol. 10, no. 4, p. e1003544, 2014.

[43] M. W. Sneddon, J. R. Faeder, and T. Emonet, "Efficient modeling, simulation and coarse-graining of biological complexity with NFsim," *Nat. Methods*, vol. 8, no. 2, pp. 177–183, 2011.

[44] U. Wittig, R. Kania, M. Golebiewski *et al.*, "SABIO-RK–database for biochemical reaction kinetics," *Nucleic Acids Res.*, vol. 40, no. Database issue, pp. D790–D796, 2012.

[45] UniProt Consortium, "UniProt: a hub for protein information," *Nucleic Acids Res.*, vol. 43, no. Database issue, pp. D204–D212, 2015.

[46] P. D. Karp, S. M. Paley, M. Krummenacker *et al.*, "Pathway Tools version 13.0: integrated software for pathway/genome informatics and systems biology," *Brief. Bioinform.*, vol. 11, no. 1, pp. 40–79, 2010.

2015 Whole-Cell Modeling Summer School included the 54 participants listed in Table SI in the supplemental material.