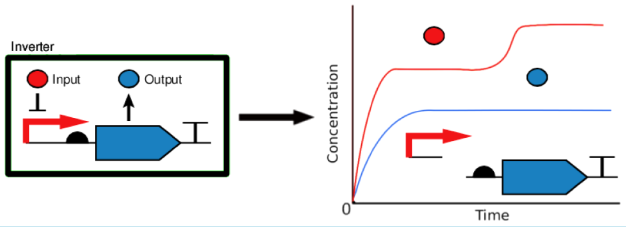
Generating Systems Biology Markup Language Models from the Synthetic Biology Open Language

Nicholas Roehner,*,[†] Zhen Zhang,[‡] Tramy Nguyen,[‡] and Chris J. Myers[‡]

[†]Department of Bioengineering, University of Utah, Salt Lake City, Utah 84112, United States

[‡]Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, Utah 84112, United States

Supporting Information



ABSTRACT: In the context of synthetic biology, model generation is the automated process of constructing biochemical models based on genetic designs. This paper discusses the use cases for model generation in genetic design automation (GDA) software tools and introduces the foundational concepts of standards and model annotation that make this process useful. Finally, this paper presents an implementation of model generation in the GDA software tool iBioSim and provides an example of generating a Systems Biology Markup Language (SBML) model from a design of a 4-input AND sensor written in the Synthetic Biology Open Language (SBOL).

KEYWORDS: *SBOL, SBML, standards, modeling, genetic design automation*

Biochemical modeling is the process of constructing a mathematical description of a biochemical system. Within the fields of systems and synthetic biology, a great deal of effort has gone into the development of software tools[1−6] for biochemically modeling both natural and synthetic biological systems. These tools have enabled their users to construct, simulate, and otherwise analyze biochemical models of biological systems, with applications ranging from hypothesis testing to engineering design.

The latter application has been particularly emphasized in synthetic biology, where many biochemical modeling tools[7−15] have been developed for the purpose of genetic design automation (GDA).[16] These tools typically automate the process of generating quantitative models from designs that contain data on the structure and function of genetic components, such as component types, genetic sequences, regulatory interactions, and performance measurements. Most of these tools generate composite models based on the composition of previously constructed models for genetic components or larger circuits, but some tools, such as SynBioSS[8] and GenoCAD,[12,17] include the ability to generate models based on components imported from the Registry of Standard Biological Parts.[18] None of these tools, however, generate models from a standardized domain-specific language for genetic designs. It is this process of model generation from a standardized language, and its specific implementation in the GDA software tool iBioSim,[19] that form the subject of this paper.

Besides saving time through automation, model generation tools can be used to facilitate interdisciplinary collaboration. For instance, these tools can be used by a synthetic biologist who lacks expertise in applied mathematics to generate models based on his or her genetic designs. The generated models can then be analyzed by an engineer whose background is in mathematics as opposed to biology. In this way, the audience for a biologist's qualitative designs is expanded to include an engineer who is interested in quantitative models.

In addition, the use of model generation tools can facilitate the comparison of models from different research groups. Since the models generated by these tools are based on genetic designs, there exist specific mappings from a given design to the models generated from the design. If these mappings are documented during the model generation process, then it becomes possible to more formally compare different models that are generated from the same genetic design.

These collaborative benefits, however, are not possible without the incorporation of standards for genetic design and model annotation into model generation tools. The rest of the introduction briefly introduces these concepts and the roles that they play in model generation.

Standards. There are at least two qualities of good standards that make model generation tools possible and enhance their efficacy. The first of these qualities is data restriction. Because standards require that data be written in a restricted format, it becomes possible for model generation tools to leverage well-defined mappings between standards for genetic design and biochemical modeling. The second quality of good standards is openness, which has implications for the uptake of standards within a community. When standards are free for use by all interested parties, it is easier for these standards to spread and gain traction within the community as

Special Issue: IWBDA 2014

Received: September 30, 2014
Published: March 30, 2015

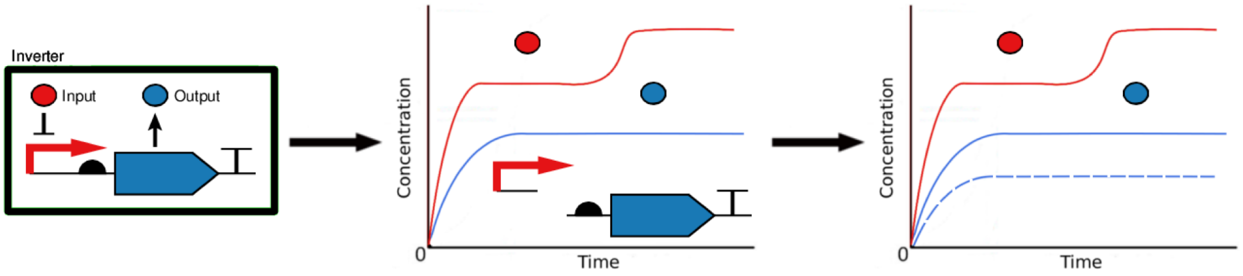


Figure 1. Model generation process in iBioSim. The structure of the SBOL module definition for a genetic inverter (left) informs the construction of a SBML model annotated with SBOL elements (middle). Following the construction and annotation of the model, its default parameters must be tuned or fitted to experimental data to produce desired behaviors during simulation (right). All DNA component symbols in this and subsequent figures adhere to the SBOL Visual standard[31] and have been created using the Pigeon web application.[32]

a whole. Widespread use of standards enhances the collaborative benefits of model generation tools because it increases the number of potential users for these tools and the size of the potential audience for users' generated models.

While most of the model generation tools for synthetic biology support export of standardized models, most notably Systems Biology Markup Language (SBML)[20,21] models, far fewer support import of standardized genetic designs, such as those written in the Synthetic Biology Open Language (SBOL).[22,23] In the case of SBOL, this is partly due to the fact that the standard has lacked the means to represent the function of genetic designs in addition to their structure. Recently, however, a proposal has been made to supplement the genetic components and sequences of SBOL with module definitions that group functional components on the basis of their intended function and assert the component interactions required for this function.[24]

As standards for genetic design continue to develop and incorporate engineering concepts for design organization, such as hierarchy and modularity, it becomes necessary for model generation tools to support these standards and map designs written in them to quantitative modeling standards that are capable of encoding similar concepts. Without the use of standards that encode hierarchy and modularity, different model generation tools cannot exchange information on the higher-order organization of models. To address this problem, the model generation methodology presented in this paper focuses on generating hierarchical SBML models[25] from genetic designs that adhere to an updated proposal for the next version of SBOL.

Model Annotation. The primary use of model annotation is to associate the elements of a model with data that cannot otherwise be represented in the format or standard in which the model is written. In the context of model generation for synthetic biology, model annotation can be thought to have at least two use cases.

In the first use case, the purpose of a model annotation is to indicate the provenance or origin of a generated model element. For example, when generating SBML from SBOL, the species and reactions of a SBML model can be annotated with SBOL elements from the functional layer of design, such as the functional components within a SBOL module definition and the interactions in which they participate. In this way, a record is kept of which elements in a generated biochemical model correspond to which elements in its source genetic design, which can later be used to compare models generated from the same design.

In the second use case, the purpose of a model annotation is to indicate the molecular identity of a model element that represents or is encoded by a biochemical structure. For example, the species of a SBML model can be annotated with SBOL elements from the structural layer of design, such as component definitions for protein, RNA, and DNA components that link to their genetic sequences. This data can be particularly useful for matching molecular inputs and outputs or avoiding cross-talk when composing models for genetic circuits or metabolic pathways. In addition, this data can be used by sequence generation tools to infer composite genetic sequences from the cause-and-effect organization of the annotated model.[26,27]

The model generation methodology presented in this paper annotates generated SBML with SBOL to accomplish both of these use cases. In addition, since the elements of SBOL include terms from ontologies, this SBML-to-SBOL annotation scheme adheres as close as possible to the standard set forth by the MIRIAM project[28] for annotating systems biology models. An ontology is controlled vocabulary that defines terms as well as the relationships between terms. Ontologies used by SBOL include the Sequence Ontology,[29] for providing roles in genetic component definitions (such as promoter and terminator) and the Systems Biology Ontology (SBO),[30] for providing types of biochemical interactions and roles in these interactions (such as activation, repression, activator, and repressor).

■ RESULTS AND DISCUSSION

Figure 1 illustrates the process of model generation in iBioSim. As represented by the first arrow in this diagram, model generation is the construction of a quantitative SBML model based on the structure and content of a qualitative SBOL module definition. During this construction, the biochemical species and reactions of the generated SBML model are also annotated with elements from the SBOL module definition in order to document the provenance of the model and the molecular identities of its species. Finally, as represented by the second arrow, the parameters of the generated SBML model can be tuned or fitted to experimental data to produce accurate behaviors during simulation.

Since the proposal for the next version of SBOL is not capable of encoding quantitative parameters, the rate laws of the SBML reactions generated by iBioSim are populated with default parameters that must be customized using iBioSim or another SBML-compatible modeling tool. Table 1 lists these default parameters and their current values. In the future, SBOL could be extended with the capacity to store data on quantitative parameters and measurements. Storing this

Table 1. Default Parameters for Rate Laws of Generated SBML Reactions

parameter	symbol	value	units
rate of degradation	k_d	0.0075	1/s
stoichiometry of production	n_p	10	unitless
open complex production rate	k_o	0.05	1/s
basal production rate	k_b	0.0001	1/s
activated production rate	k_a	0.25	1/s
promoter count	n_g	2	molecule
RNA polymerase binding equilibrium	K_o	0.033	1/molecule
activated RNA pol. binding equilibrium	K_{oa}	1	1/molecule
RNA polymerase count	n_r	30	molecule
repression binding equilibrium	K_r	0.5	1/molecule
activation binding equilibrium	K_a	0.0033	1/molecule
stoichiometry of binding	n_c	2	unitless
forward noncovalent binding rate	k_{c_f}	0.05	1/(molecule·sec)
noncovalent binding equilibrium	K_c	0.05	1/molecule
reverse noncovalent binding rate	k_{c_r}	1	1/sec

information would provide a firmer foundation for GDA tools to generate different mathematical models for different design tasks that nevertheless conform to the same basic data set.

The derivation of the rate laws for the reactions generated by iBioSim is based on the application of the law of mass action and model abstraction techniques, such as operator site reduction[33,34] and quasi-steady-state approximation.[35,36] While a more detailed description of this type of derivation can be found in the literature,[37] a short summary is included here. Briefly, this derivation assumes that the noncovalent binding of transcription factors (TFs) to DNA occurs on a faster time scale than transcription, translation, and protein degradation, such that the complexes formed between TF species and DNA are at or near their equilibrium levels at all times. This assumption is typically made to simplify genetic circuit models for more tractable analysis and efficient simulation. As a consequence of this assumption, the models generated by iBioSim do not include reactions for TF species binding to DNA, nor do they include the complexes between TFs and DNA that are produced by these reactions. Instead, these complexes are replaced in the mass conservation laws for each promoter with the product of the amounts for the promoter, its binding TF species, and equilibrium binding constant. This allows the rate of genetic production (transcription and translation) from each promoter to be rewritten as a fraction in which each term of the denominator and numerator represents a different state of the promoter. As seen in Figure 2, when the amount of a repressor species increases, the denominator increases and the rate of genetic production from the promoter is eventually minimized. When the amount of an activator species increases, however, both the numerator and denominator increase and the rate of genetic production from the promoter is eventually maximized.

Given that the form of any model is generally only valid under certain conditions, model generation tools need to be flexible enough to abstract between different forms of model as circumstances dictate. iBioSim implements a range of techniques for automated model reduction and expansion that may be applied to a generated model to either increase or decrease its level of abstraction. For example, if a user decides that the noncovalent binding between TF species and small molecule inducer species occurs on the same fast time scale as TF species binding to DNA, then iBioSim can eliminate the relevant reactions and add equivalent algebraic expressions to the rate laws of the appropriate genetic production reactions.[34]

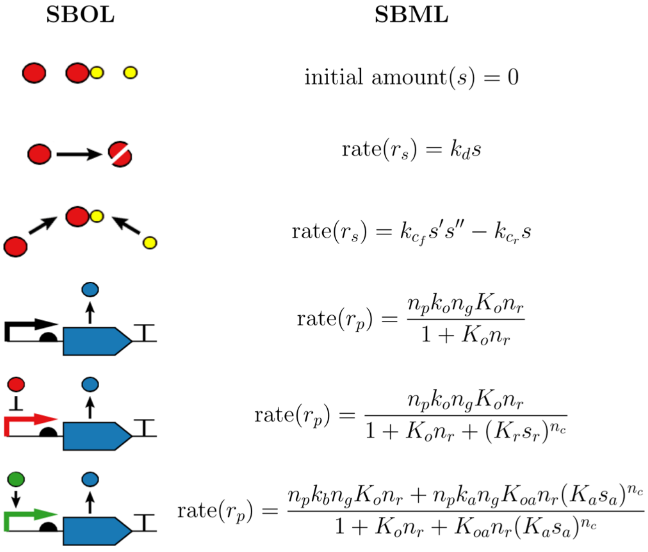

SBOL
SBML
initial amount$(s) = 0$
$\text{rate}(r_s) = k_d s$
$\text{rate}(r_s) = k_{c_f} s' s'' - k_{c_r} s$
$\text{rate}(r_p) = \dfrac{n_p k_o n_g K_o n_r}{1 + K_o n_r}$
$\text{rate}(r_p) = \dfrac{n_p k_o n_g K_o n_r}{1 + K_o n_r + (K_r s_r)^{n_c}}$
$\text{rate}(r_p) = \dfrac{n_p k_b n_g K_o n_r + n_p k_a n_g K_{oa} n_r (K_a s_a)^{n_c}}{1 + K_o n_r + K_{oa} n_r (K_a s_a)^{n_c}}$

Figure 2. Mapping from SBOL to the initial amounts and rate laws for generated SBML species and reactions, respectively. Each protein, complex, or small molecule (top) is mapped to a species with an initial amount of zero. If a protein, complex, or small molecule is degraded (second from top), then its corresponding species is assigned as a reactant for a reaction with no products. By the law of mass action, the rate law for this degradation reaction is first order. Noncovalent bindings that include a complex (third from top) are mapped to a reversible reaction with the appropriate complex species as its product and the TF or inducer species that form the complex as its reactants. When the amounts of the latter species are increased, the rate of complex formation is increased until an equilibrium is achieved with the negative rate law term for the reverse reaction. Finally, all constitutively expressed, repressed, and activated genes (bottom three) are mapped to reactions that only have products. These genetic production reactions are modified by the appropriate TF species and have more complex fractional rate laws resulting from operator site reduction. The rate laws for constitutive and repressed genes capture the ratio of RNA polymerase-bound promoter states to all other states, while those for activated genes capture the ratio of polymerase-bound and activator-polymerase-bound states to all other states. These ratios are then multiplied by the gene copy number (promoter count), equilibrium binding of RNA polymerase, and maximum rate of transcription initiation (open complex production) to determine the rate of genetic production.

One immediate application of model generation in iBioSim is the creation of libraries of simulatable genetic designs that can serve as inputs to genetic technology mapping. Genetic technology mapping is the process of selecting genetic components from a library to meet an abstract behavioral specification, an important application of GDA tools.[9,10,13−15,38,39] In iBioSim, this process involves automatically selecting one or more SBOL-annotated SBML models from a design library and composing them to optimally satisfy a SBML model specification for an abstract genetic circuit.[40] Thanks to model generation, the concrete genetic circuit design resulting from genetic technology mapping contains a composite SBML model that can be simulated to verify that the theoretical function of the design is correct.

Example: 4-Input AND Sensor. In order to demonstrate the scale and potential application of model generation in

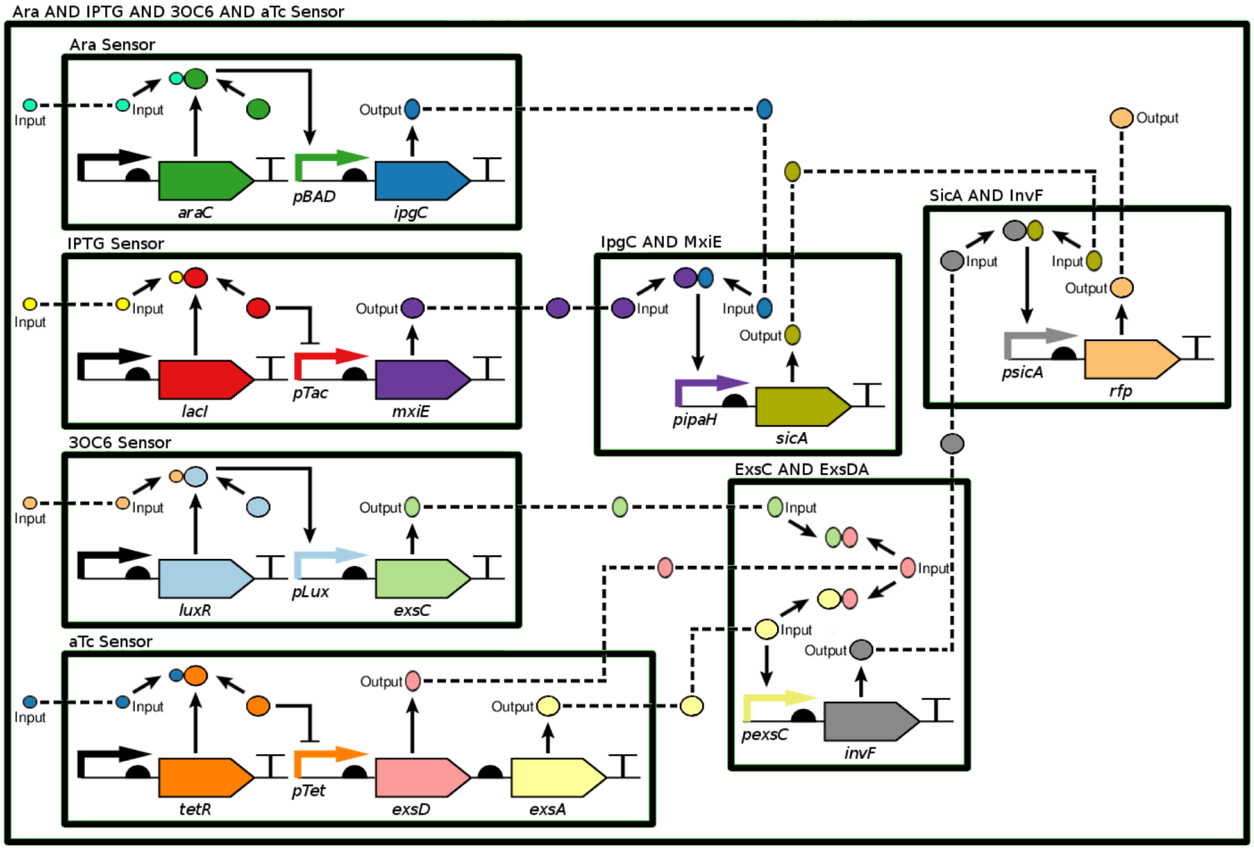


Figure 3. Visualization of the SBOL for the 4-input AND sensor. Not pictured is the degradation of each noninput component. Each sensor constitutively expresses a TF that interacts with a small molecule inducer. In the case of the Ara and 3OC6 sensors, the inducer enables the TF to activate production of the output protein. For the IPTG and aTC sensors, however, the inducer sequesters the TF and relieves repression of output protein production. When all four inducers are present, the sensors produce a combination of chaperone and TF proteins. These proteins then bind to form complexes that either activate production, as in the case of the IpgC-MxiE AND gate, or alternately sequester an activator TF and relieve its sequestration, as in the case of the ExsC-ExsDA AND gate. Finally, these AND gates produce a chaperone and a TF, respectively, that form a complex and activate production of RFP from the SicA-InvF AND gate.

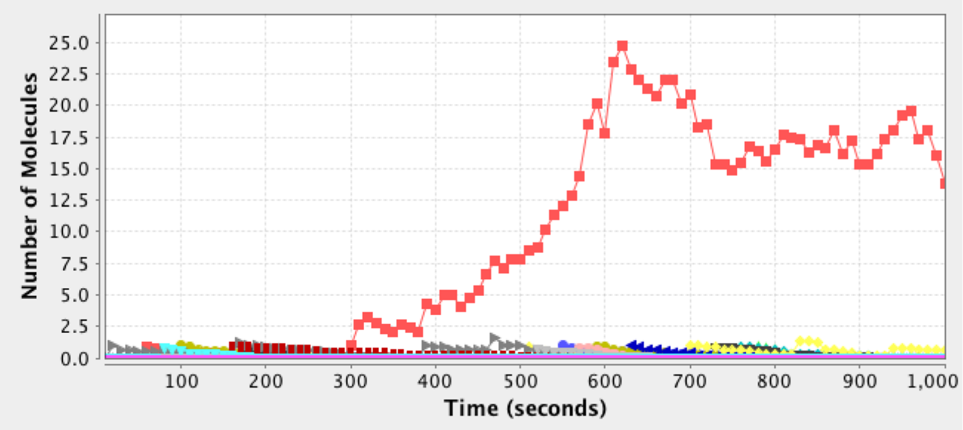


Figure 4. Simulation results for the 4-input AND sensor. Each time series is the average of 10 stochastic simulation results performed using the Gillespie algorithm[43] and represents the amount of RFP produced for a particular combination of low and high inducer inputs, where low is zero molecules of inducer and high is 60 molecules. The red time series is the amount of RFP produced when all four inducer inputs are high.

iBioSim, this section presents an example of generating a SBML model from the SBOL module definitions for four sensors connected to a 4-input AND gate, which is among the largest genetic circuits ever constructed.[41] The module definitions shown in Figure 3 differ slightly from their original conception in ref 41, in that each module definition is organized to have proteins as its input and outputs, rather than transcription signals of RNA polymerases per second. The former is more conducive to generating the form of genetic circuit model that iBioSim is best suited to analyze, one in which submodels communicate *via* varying amounts of protein. In principle, however, the behavior resulting from a polymerase operations per second (PoPs) model is equivalent. In the future, iBioSim could be extended to provide advanced analysis options, such as Markov chain analysis,[42] for both types of model.

Figure 4 displays the simulation results for the SBML model generated from the SBOL module definitions describing the 4-input AND sensor. Each time series in these results is the average of 10 stochastic simulation runs and captures the behavior of the AND sensor in response to a different possible combination of low and high inducer inputs. As expected, there is an order of magnitude difference in expression between the case when all inducer inputs are high and all other cases. This is not dissimilar from the experimental results in ref 41, though the expression for some low input combinations is less leaky in the model. Greater agreement and predictive power could be obtained by independently measuring the parameters of the model or fitting them to a subset of the relevant experimental data. Ultimately, these results show that model generation and simulation in iBioSim can be used to reproduce behavior observed in the lab for recent artificial gene networks.

Discussion. The model generation methodology presented in this paper enables users of iBioSim to generate quantitative SBML models from qualitative SBOL module definitions. Furthermore, the generated SBML models are annotated with their source SBOL in order to tightly couple these quantitative and qualitative descriptions of genetic function and structure. The end result is a library of simulatable genetic designs that have the potential to be exchanged between standard-compliant GDA software tools.

It is important to note, however, that this approach to model generation captures only one of many possible mappings between SBOL and various standards for biochemical modeling. In particular, this approach specifies a mapping between SBOL and a specific form of genetic circuit model written in SBML, one that is suited to simulation as a system of ordinary differential equations (ODEs) or stochastic processes. SBML, however, is capable of specifying other types of models, such as algebraic models for the analysis of metabolic pathways. In addition, there exist other modeling standards besides SBML, including those expressly developed for modeling biology, such as CellML,[44] and scripting/programming languages commonly applied to modeling biology, such as MATLAB[45] and Python.

In time, model generation must grow to accommodate other possible mappings between standards for genetic design (most notably SBOL) and biochemical modeling. In order to facilitate this growth, further research is necessary to build upon previous model generation formalisms, such as the grammars used by GenoCAD.[12] Products of this research would include methods for automatically comparing biochemical models written in different standards, provided that these models are generated from the same genetic design. GDA tools could be developed that enable users to create and store new mappings from SBOL to different biochemical modeling standards. Tools and methods such as these would help to democratize model generation and involve larger segments of the synthetic biology community in its long-term growth.

■ METHODS

The procedure for automatically generating annotated SBML from SBOL follows the steps outlined below. Note that the expected SBOL conforms to a recently proposed data model.[24] This data model has undergone some minor revisions that will be detailed in the forthcoming specification for the next version of SBOL. For the purposes of model generation, the most significant revisions are that the Module and ModuleInstantiation classes have been renamed to ModuleDefinition and Module, respectively, the Component class has been renamed to ComponentDefinition, and the ComponentInstantiation class has been renamed to ComponentInstance and subclassed as FunctionalComponent and Component. The latter subclasses denote whether a component belongs to a module (the functional layer of design) or to another component (the structural layer of design).

In addition, the Port class has been replaced with data fields on the ComponentInstance and MapsTo (formerly PortMap) classes. These data fields include an *access* field that specifies whether a component instance can be mapped to another component instance (*public* or *private*), a *direction* field that specifies whether a component instance is an *input*, *output*, both, or neither, and a *refinement* field that provides additional semantics for mappings between component instances. For example, a mapping may use its refinement field to indicate that one component instance should be identical to another (*verifyIdentical*), that one instance takes precedence over another (*useLocal* or *useRemote*), or that two instances should be considered in combination (*merge*).

1. For each SBOL module definition in a SBOL document:
   (a) Add a SBML model to a new SBML document.
   (b) Annotate the SBML model with the SBOL module definition.
   (c) Follow steps 2 through 6.
2. For each functional component i in the SBOL module definition that is a protein, small molecule, or complex:
   (a) Add a species s to the list of species in the SBML model.
   (b) Annotate s with i and the component definition for i.
   (c) If the direction of i is set to "input" or "output", add a port to the SBML model, set its ID reference to s, and label it with the SBO term "input port" or "output port".
   (d) If the direction of i is set to "input", mark s as a boundary condition.
   (e) If i is the sole degraded participant in a single degradation interaction n:
      i. Add a reaction r_s to the list of reactions in the SBML model and label it with the SBO term "degradation".
      ii. Annotate r_s with n.
      iii. Add a species reference e for s to the list of reactants for r_s.
      iv. Annotate e with the participation of i in n.
      v. Add a kinetic law of the form below to r_s.

$$\text{rate}\ (r_s) = k_d s \tag{1}$$

3. For each functional promoter DNA component i:
   (a) Add a species p to the list of species in the SBML model.
   (b) Annotate p with i and the component definition for i.
   (c) Add a reaction r_p to the list of reactions in the SBML model and label it with the SBO term "genetic production".
   (d) Add a modifier species reference e for p to the list of modifiers for r_p and label it with the SBO term "promoter".

   (e) For each genetic production interaction n in which i participates as a promoter, a functional protein component j participates as a product, and a functional gene DNA component k is a transcribed participant:
      i. Annotate r_p with n.
      ii. Annotate e with the participation of i in n.
      iii. Add a species reference e' for the species s that corresponds with j to the list of products for r_p.
      iv. Annotate e' with the participation of j in n.
      v. Annotate s with the component definition for k.
   (f) For each activation or repression interaction n' in which i is a repressed or activated participant and a functional TF protein component x participates as an activator or repressor:
      i. Annotate r_p with n'.
      ii. Annotate e with the participation of i in n'.
      iii. Add a modifier species reference e' for the species y that corresponds with x to the list of modifiers for r_p and label it with the SBO term “repressor” or “activator.”.
      iv. Annotate e' with the participation of x in n'.
      v. Add y to the set of activators $\mathrm{Act}(p)$ or set of repressors $\mathrm{Rep}(p)$.
   (g) If r_p has no products, remove it from the SBML model; otherwise, add a kinetic law of the form below to r_p.

$$\text{rate }(r_p) = \begin{cases} \dfrac{n_p k_o n_g K_0 n_r}{1 + K_0 n_r + \sum_{s_r \in \mathrm{Rep}(p)} (K_r s_r)^{n_c}} & \mathrm{Act}(p) = 0 \\ \dfrac{n_p k_b n_g K_0 n_r + n_p k_a n_g K_{0a} n_r \sum_{s_a \in \mathrm{Act}(p)} (K_a s_a)^{n_c}}{1 + K_0 n_r + \sum_{s_r \in \mathrm{Rep}(p)} (K_r s_r)^{n_c} + K_{0a} n_r \sum_{s_a \in Act(p)} (K_a s_a)^{n_c}} & \text{otherwise} \end{cases} \tag{2}$$

4. For each noncovalent binding interaction n in which a functional complex component i participates as a complex and a set $\mathrm{Comp}(i)$ of one or more functional small molecule or protein components participate as ligands:
   (a) Add a reversible reaction r_s to the list of reactions in the SBML model, where s is the species that corresponds with i, and label it with the SBO term “noncovalent binding”.
   (b) Annotate r_s with n.
   (c) Add a species reference e for s to the list of products for r_s.
   (d) Annotate e with the participation of i in n.
   (e) Add a set of species references $\mathrm{ref}(s)$ to the list of reactants for r_s, where each species reference is for a species in the set $\mathrm{React}(s)$ that corresponds with $\mathrm{Comp}(i)$.
   (f) Annotate each species reference in $\mathrm{ref}(s)$ with the corresponding participation in n.
   (g) Add a kinetic law of the form below to r_s.

$$\text{rate }(r_s) = k_{c_f} K_c^{|\mathrm{React}(s)|-2} \prod_{s' \in React(s)} s' - k_{c_r} s \tag{3}$$

5. For each submodule u:
   (a) Add a submodel v to the SBML model.
   (b) Annotate v with u.
   (c) Add an external model definition for the SBML model that corresponds with the module definition for u to the list of external model definitions in the SBML model.
   (d) For each maps-to element t of u that maps between a local functional component j in this module definition and a remote functional component k in the module definition for u:
      i. If the refinement of t is set to “verify identical”, “use local”, or “merge”, create a replaced element q; otherwise, if this refinement is set to “use remote”, create a replaced-by element q.
      ii. If q is a replaced-by element, add it to the species that corresponds with j; otherwise, add it to the list of replaced elements for that species.
      iii. Set the submodel reference for q to v and its port reference to the port for the species that corresponds with k.
      iv. Annotate q with t.

■ ASSOCIATED CONTENT

Ⓢ Supporting Information

Supporting Information includes a zip archive that contains an iBioSim project file and PDF instructions for generating SBML from SBOL in iBioSim. The iBioSim project file includes a SBOL file that contains the module and component definitions for the 4-input AND sensor and the annotated SBML model files generated from this SBOL file. This material is available free of charge *via* the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: n.roehner@utah.edu.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. CCF-1218095 and Grant No. DBI-1356041. 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 National Science Foundation.

■ REFERENCES

(1) Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P., and Kummer, U. (2006) COPASI: a COmplex PAthway SImulator. *Bioinformatics 22*, 3067−3074.

(2) Drager, A., Hassis, N., Supper, J., Schroder, A., and Zell, A. (2008) SBMLsqueezer: a CellDesigner plug-in to generate kinetic rate equations for biochemical networks. *BMC Syst. Biol. 2*, 39.

(3) Funahashi, A., Matsuoka, Y., Jouraku, A., Morohashi, M., Kikuchi, N., and Kitano, H. (2008) CellDesigner 3.5: a versatile modeling tool for biochemical networks. *Proc. IEEE 96*, 1254−1265.

(4) Chandran, D., Bergmann, F. T., and Sauro, H. M. (2009) TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng. 3*, 19.

(5) Mirschel, S., Steinmetz, K., Rempel, M., Ginkel, M., and Gilles, E. D. (2009) ProMoT: modular modeling for systems biology. *Bioinformatics 25*, 687−689.

(6) Wrzodek, C., Drager, A., and Zell, A. (2011) KEGGtranslator: visualizing and converting the KEGG PATHWAY database to various formats. *Bioinformatics 27*, 2314−2315.

(7) Rodrigo, G., Carrera, J., and Jaramillo, A. (2007) Asmparts: assembly of biological model parts. *Syst. Synth. Biol. 1*, 167−170.

(8) Hill, A. D., Tomshine, J. R., Weeding, E. M., Sotirpoulos, V., and Kaznessis, Y. N. (2008) SynBioSS: the synthetic biology modeling suite. *Bioinformatics 24*, 2551−2553.

(9) Marchisio, M. A., and Stelling, J. (2008) Computational design of synthetic gene circuits with composable parts. *Bioinformatics 24*, 1903−1910.

(10) Pedersen, M., and Phillips, A. (2009) Towards programming languages for genetic engineering of living cells. *J. R. Soc., Interface 6*, S437−S450.

(11) Czar, M. J., Cai, Y. Z., and Peccoud, J. (2009) Writing DNA with GenoCAD^{TM}. *Nucleic Acids Res. 37*, W40−W47.

(12) Cai, Y., Lux, M. W., Adam, L., and Peccoud, J. (2009) Modeling structure−function relationships in synthetic DNA using attribute grammars. *PLoS Comput. Biol. 5*, e1000529.

(13) Beal, J., Lu, T., and Weiss, R. (2011) Automatic compilation from high-level biologically-oriented programming language to genetic regulatory networks. *PLoS One*, DOI: 10.1371/journal.pone.0022490.

(14) Huynh, L., Tsoukalas, A., Koppe, M., and Tagkopoulos, I. (2013) SBROME: a scalable optimization and module matching framework for automated biosystems design. *ACS Synth. Biol. 2*, 1073−1089.

(15) Marchisio, M. (2014) Parts & Pools: a framework for modular design of synthetic gene circuits. *Front. Bioeng. Biotechnol.*, DOI: 10.3389/fbioe.2014.00042.

(16) Lux, M. W., Bramlett, B. W., Ball, D. A., and Peccoud, J. (2012) Genetic design automation: engineering fantasy or scientific renewal? *Trends Biotechnol. 30*, 120−126.

(17) Cai, Y., Wilson, M. L., and Peccoud, J. (2010) GenoCAD for iGEM: a grammatical approach to the design of standard-compliant constructs. *Nucleic Acids Res. 38*, 2637−2644.

(18) *iGEM Registry of Standard Biological Parts*: Version 6.0.2013, http://parts.igem.org, accessed on Sept. 30, 2014.

(19) Madsen, C., Myers, C., Patterson, T., Roehner, N., Stevens, J., and Winstead, C. (2012) Design and test of genetic circuits using iBioSim. *IEEE Des. Test Comput. 29*, 32−39.

(20) Hucka, M., et al. (2003)*et al.* The Systems Biology Markup Language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 524−531.

(21) Hucka, M., Bergmann, F. T., Hoops, S., Keating, S. M., Sahle, S., Schaff, J. C., Smith, L. P., and Wilkinson, D. J. (2010) *The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core*, http://sbml.org/Documents/Specifications#SBML_Level_3_Version_1_Core, accessed on Sept. 30, 2014.

(22) Galdzicki, M., et al. (2014)*et al.* The Synthetic Biology Open Language (SBOL) provides a community standard for communicating designs in synthetic biology. *Nat. Biotechnol. 32*, 545−550.

(23) Galdzicki, M. et al.*et al.* (2012) *Synthetic Biology Open Language (SBOL)*, Version 1.1.0 BBF RFC 87, http://hdl.handle.net/1721.1/73909.

(24) Roehner, N., Oberortner, E., Pocock, M., Beal, J., Clancy, K., Madsen, C., Misirli, G., Wipat, A., Sauro, H., and Myers, C. J. (2015) Proposed data model for the next version of the Synthetic Biology Open Language. *ACS Synth. Biol. 4*, 57−71.

(25) Smith, L. P., Hucka, M., Hoops, S., Finney, A., Ginkel, M., Myers, C. J., Moraru, I., and Liebermeister, W. (2013) *Hierarchical Model Composition*, http://sbml.org/Documents/Specifications/SBML_Level_3/Packages/Hierarchical_Model_Composition_%28comp%29, accessed on Sept. 30 2014.

(26) Misirli, G., Hallinan, J. S., Yu, T., Lawson, J. R., Wimalaratne, S. M., Cooling, M. T., and Wipat, A. (2011) Model annotation for synthetic biology: automating model to nucleotide sequence conversion. *Bioinformatics 27*, 973−979.

(27) Roehner, N., and Myers, C. J. (2014) A methodology to annotate Systems Biology Markup Language Models with the Synthetic Biology Open Language. *ACS Synth. Biol. 3*, 57−66.

(28) Novere, N. L., et al. (2005)*et al.* Minimum information requested in the annotation of biochemical models (MIRIAM). *Nat. Biotechnol. 23*, 1509−1515.

(29) Eilbeck, K., Lewis, S. E., Mungall, C. J., Yandell, M., Stei, L., Durbin, R., and Ashburner, M. (2005) The Sequence Ontology: a tool for the unification of genome annotations. *Genome Biol. 6*, R44.

(30) Courtot, M., et al. (2011)*et al.* Controlled vocabularies and semantics in systems biology. *Mol. Syst. Biol. 7*, 543.

(31) Quinn, J., Beal, J., Bhatia, S., Cai, P., Chen, J., Clancy, K., Hillson, N., Galdzicki, M., Maheshwari, A., Umesh, P., Pocock, M., Rodriquez, C., Stan, G.-B., and Endy, D. (2013) *Synthetic Biology Open Language Visual (SBOL Visual)*, Version 1.0.0 BBF RFC 93, http://hdl.handle.net/1721.1/78249.

(32) Bhatia, S., and Densmore, D. (2013) Pigeon: a design visualizer for synthetic biology. *ACS Synth. Biol. 2*, 348−350.

(33) Tyson, J., and Othmer, H. G. (1978) The dynamics of feedback control circuits in biochemical pathways. *Prog. Theor. Biol. 5*, 2−62.

(34) Kuwahara, H., Myers, C., Barker, N., Samoilov, M., and Arkin, A. (2006) Automated abstraction methodology for genetic regulatory networks, in *Transactions on Computational Systems Biology VI* (Priami, C., and Plotkin, G., Eds.) pp 150−175, Springer, Berlin.

(35) Briggs, G. E., and Haldane, J. B. S. (1925) A note on the kinetics of enzyme action. *Biochem. J. 19*, 338−339.

(36) Rao, C. V., and Arkin, A. P. (2003) Stochastic chemical kinetics and the quasi-steady-state assumption: application to the Gillespie algorithm. *J. Chem. Phys. 118*, 4999−5010.

(37) Myers, C. J. (2009) *Engineering Genetic Circuits*, Chapman and Hall/CRC, Boca Raton, FL.

(38) Yaman, F., Bhatia, S., Adler, A., Densmore, D., and Beal, J. (2012) Automated selection of synthetic biology parts for genetic regulatory networks. *ACS Synth. Biol. 1*, 332−344.

(39) Huynh, L., and Tagkopoulos, I. (2014) Optimal part and module selection for synthetic gene circuit design automation. *ACS Synth. Biol. 3*, 556−564.

(40) Roehner, N., and Myers, C. J. (2014) Directed acyclic graph-based technology mapping of genetic circuit models. *ACS Synth. Biol. 3*, 543−555.

(41) Moon, T. S., Lou, C., Tamsir, A., Stanton, B. C., and Voigt, C. A. (2012) Genetic programs constructed from layered logic gates in single cells. *Nature 491*, 249−253.

(42) Madsen, C., Myers, C. J., Roehner, N., Winstead, C., and Zhang, Z. (2012) Utilizing stochastic model checking to analyze genetic circuits. *Proc. IEEE Symp. Comput. Intell. Bioinf. Comput. Biol.*, 379−386. San Diego, CA.

(43) Gillespie, D. T. (1977) Exact stochastic simulation of coupled chemical kinetic reactions. *J. Phys. Chem. 81*, 2340−2361.

(44) Hedley, W. J., Nelson, M. R., Bellivant, D. P., and Nielsen, P. F. (2001) A short introduction to CellML. *Philos. Trans. R. Soc., A 359*, 1073−1089.

(45) (2014) *MATLAB*, version 8.3 (R2014a), The MathWorks, Inc., Natick, MA.