I. SynBioSS Wiki

Introduction to SynBioSS Wiki

SynBioSS Wiki is a component of the Synthetic Biology Software Suite (synbioss.sourceforge.net). SynBioSS Wiki is practically two things: i) a web interface based on the MediaWiki package and ii) a database for storing molecular components, their interactions and pertinent biological information. The purpose of SynBioSS Wiki is to enable the scientific community to store and retrieve information related to synthetic biology efforts, and to facilitate the creation of networks of biochemical reactions that can be modeled by the SynBioSS Desktop Simulator (DS). This document is meant to serve as a detailed description of the SynBioSS Wiki project in terms of the front-end user experience and the underlying database, programming, and extensions to the MediaWiki software.

Database Structure

The SynBioSS Wiki goes beyond the MediaWiki software in storing kinetic information in a formatted (and therefore machine-searchable) format. In general, MediaWiki is designed to store "articles," as exemplified by the format of Wikipedia entries. While such articles may contain a wealth of information, this information is formatted for interpretation by a human reader rather than an automated algorithm. A database of kinetic constants, on the other hand, should be easily searchable by participating species, reaction type, etc.

The underlying SynBioSS Wiki database, called "biowikidb", contains two sections. The first section is the standard MediaWiki tables, necessary for the MediaWiki software to function properly. These tables have not been edited, and thus are fully documented on the MediaWiki website [1]. Note that for this project, each MediaWiki table has a prefix of "bw_", i.e. the standard table "archive" is "bw_archive" in our database.

The other section of the database contains the tables generated specifically for the SynBioSS Wiki project. A graphical representation is provided below, with all primary keys in bold text:

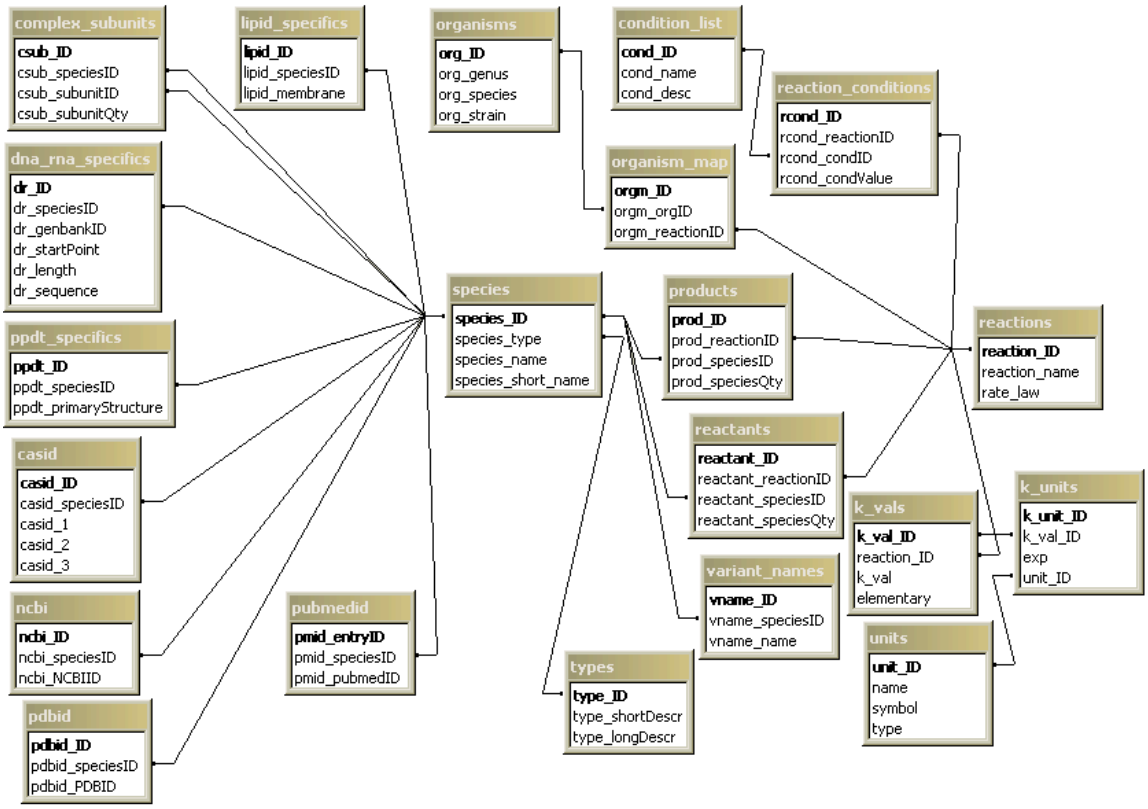


Figure 1 – Layout of the database supporting SynBioSS Wiki (excluding the tables normally associated with the MediaWiki software).

Many tables & fields are self explanatory, although certain fields merit clarification.

Table 1 – Descriptions of selected tables and fields depicted in Figure 1.

Table	Column	Description
casid	casid_1	CAS ID numbers are composed of three series of digits separated by dashes, e.g. 1234-56-7. The "casid_1" in this case is "1234".
casid	casid_2	casid_2 is "56" for previous example
casid	casid_3	casid_3 is "7" for previous example.
complex_subunits	csub_speciesID	The ID of the complex itself.
complex_subunits	csub_subunitID	The ID of a species within the complex.
k_vals	elementary	Equal to either 1 or 0. If "1", the reaction the k value refers to has an elementary rate law.
reactions	rate_law	If the reaction's rate is non-elementary, the complex rate law is stored here as MathML.
units	name	The full name for the unit, e.g. "second"

units	symbol	The symbol for the unit, e.g. “s”
units	type	Categories such as mass, volume, length.

MediaWiki Software and Web Interface

Licensed under the GNU General Public License, MediaWiki is a free software package. It is written in PHP and requires a database behind it; it may use either a MySQL or PostgreSQL database management system. The remainder of this document details how the MediaWiki package was used to develop a web interface to facilitate the creation of networks of biochemical reactions. These models may then be analyzed in any external simulation software of the user’s choice.

The wiki’s root directory is /var/www/wiki/. All files and directories referenced hereafter are relative to this root directory, i.e. the “includes” directory is /var/www/wiki/includes/.

A short overview of the languages used in this project follows.

- **PHP:** The main programming language used. PHP is a scripting language interpreted and executed on the web server.
- **SQL:** The query language used for communication between the PHP-scripted web pages and the database.
- **HTML:** Used to format how the wiki displays in a web browser. HTML and PHP often co-exist in the same files, though the PHP code is hidden from the end-user.
- **Javascript:** A scripting language embedded in HTML documents and interpreted/executed locally by the user’s browser.
    - **AJAX:** Technique used to make the interface more user-friendly. More specifically, on the “Add a Species” and “Add a Reaction” pages, one may search for a species in the database without reloading the entire page.
    - **DHTML:** Utilized for similar reasons as AJAX. On the “Add a Species” page, once the user selects a type, the remainder of the form appears automatically. The page must reload, however.
- **XML**

- **MathML:** Standardized format for storing mathematical expressions. Used to represent complex rate laws.
- **SBML:** Standardized format for representing networks of biochemical reactions. The desired output format for models created by the user via the wiki.

Custom Pages

A list of the currently used Special Pages is below. The name of the page is in bold, while its corresponding filename in both the /includes/ and /extensions/ directory is in parentheses (php files of identical names but different content exist in both directories simultaneously).

Add a Species *(SpecialSpecies.php)*

This page allows the user to add a new species to the database. Upon entering the name of the species, its PubMedID, its NCBI ID, and selecting its type, he/she may then enter further information contingent upon the type:

- **Complex:** A complex is any species composed of two or more simpler species bound together. Each constituent species must exist within the database separately. The name and quantity of species within the complex must be specified. This is done using the AJAX search function described earlier.
- **DNA/RNA:** Start point, length, sequence, GenBankID
- **Lipid:** Membrane specifics
- **Protein:** Primary structure, PDB ID
- **Small Molecule:** CAS registry number

Upon submitting a species, the user is redirected to the new page for that species. Specific information about these individual pages is outlined in the "Edited Pages" section.

Add/Edit a Reaction *(SpecialReaction.php)*

Depending on the way in which this page is accessed, a user may either add a new reaction or edit an existing one.

In the former case, the user must first use the AJAX search function (outlined earlier) to select a species, enter its stoichiometry, and indicate whether it should be a reactant or a product. In general, catalysts are both reactants and products, and would be entered on both sides of the equation. Upon entering at least one reactant or product, the user's reaction will appear on the page. If desired they may select one or more species in the reaction and remove it, or clear the entire reaction.

Following the specification of reaction stoichiometry, the form provides fields for kinetic data, contingent upon the complexity of the reaction's rate law.

- **Elementary:** Add the kinetic constant and its units. The user need not enter the specific elementary rate used, as this is self-evident in his/her choice of units and reaction stoichiometry.
- **Arbitrary:** The user may upload a file containing content-MathML describing the equation for the rate law. The MathML is parsed and its parameters are displayed on the Add a Reaction page. The user may then specify information for each parameter. If the parameter is a constant, the user may enter its numerical value and units (if applicable). He/She may also choose from a list of common constants, such as Boltzmann's, if so desired. If the parameter is a variable, the user shall supply a brief description of what it represents (e.g. "Concentration of species A") and the units in which this value should be entered. Parameters whose values have been specified are displayed on the page for future reference.
-

If a user visits an individual species' page and selects a reaction to edit, they will be directed to this page again, where they may edit all aspects of the reaction.

Custom Model *(SpecialModel.php)*

On this page the user may view the reactions he/she has added to his/her model. The reactions and their corresponding kinetic data are displayed. The user may delete one or many reactions from his/her model or clear the whole model if desired. At the bottom of the page is a submit button which converts the user's model seen on the wiki into

formatted SBML which the user may then save. This SBML file may be imported into any number of simulation programs and further analyzed.

Add an Organism *(SpecialOrganism.php)*

This page is used to add organisms to the database. A planned feature for the "Add a Reaction" page is to associate reactions with the organism(s) they were observed in. To do this, the user would search for organisms using an AJAX function similar to the one for species. Once again, this feature is not yet implemented, and there is currently no way for a user to view the organisms already in the database.

Edits to Existing MediaWiki Pages

The remaining SynBioSS Wiki features were implemented by editing and adding code to existing MediaWiki files; a list of these files follows. The design philosophy of the SynBioSS Wiki has centered on the creation of new pages; the editing of existing MediaWiki pages has been intentionally minimized to facilitate quick updates of the underlying MediaWiki code. A short description of the page is in bold, while the corresponding file path is in parentheses.

MediaWiki Search *(/includes/SpecialSearch.php)*

A link to the "Add a Species" page and the "Add a Reaction" page was added to the bottom of all search results. The standard MediaWiki search does not function properly with the added database tables, so the search engine has been modified. Specifically, lines 87-240 contain either edited or new code; all further expansions of the search function should be integrated there. The line `require_once("action_pages/sql.inc")` was also added to the beginning of the page, to allow for the use of the MediaWiki database functions.

MonoBook Skin *(/skins/MonoBook.php)*

One small segment was removed from immediately below line 166:

```
<input type='submit' name="go" class="searchButton" id="searchGoButton" value="<?php $this->msg('go') ?>" /> 
```

This removal causes the search bar on the left side of the wiki to display a "Search" button, but no "Go" button. Note that this change only applies to the wiki's default skin, MonoBook.

General Wiki Configuration *(/LocalSettings.php)*

A detailed explanation of wiki configuration can be found on the MediaWiki website [2]. The most notable settings adjusted for this project include the location of the database and the extensions to use. All browser caching has been deactivated, and titles of individual species pages are allowed to begin with a digit or lowercase letter.

Individual Species Pages *(/includes/Article.php)*

On each page, the species' name, PubMedID, NCBI ID, type, and additional specifics contingent on the type are listed. A list of reactions with corresponding kinetic data is displayed as well. The user may edit a reaction or add it to his/her model. Finally, by clicking the "edit" tab at the top of the page, one may add miscellaneous qualitative information about the species. Developers altering Article.php itself should add or edit code between lines 846-999, and add all `require_once()` statements to the beginning of the file.

Future Features

General

While the existing MediaWiki code and the SynBioSS Wiki extensions already compliment each other, we will continue to work to further integrate the two. For example, users should be able to view old versions of kinetic data, who edited them, and when. Clever use of this feature could alleviate concerns of data security, increase the freedom to which users can edit data, and reduce the need for manual moderation of submitted data. Currently, these features are available to users on the text-based portions of the Wiki. Reversions of kinetic data (in the case of incorrect input or intentional vandalism) are also possible, but must be effected by the database administrators.

SynBioSS Wiki / Designer Unification

Tighter integration between the SynBioSS Wiki and Designer is a high priority. As the amount of data seeding the Wiki grows, the Designer will pull kinetic data from the Wiki directly when generating models. While the small amount of data populating the Wiki currently makes such a feature "unhelpful," a populated back-end database (which the Wiki will gradually provide) will make the Designer's automatically-generated models vastly more useful by reducing or eliminating the need for manual input of kinetic parameters.

Add a Species

At present, a user must enter species information manually. Ideally, one should be able to query PubMed, GenBank, etc., from the wiki and insert the relevant information into the form automatically. This may or may not be possible with certain databases. Additionally, interaction between the wiki and BioBricks is to be implemented.

Although the table "variant_names" exists, the web interface can not yet be used to add alternative names for a species.

Add a Reaction

This page must be further adapted to the usage of MathML. While a user may already upload MathML representing a rate law, ideally one should be able to type an equation which is then converted into MathML. In addition, after the MathML is parsed, the equation itself should display on the page in a convenient graphical format. This graphical display of the equation is very close to being implemented.

Users should also be able to store information on the source of data, the conditions under which this data was obtained, the method used, and the organism(s) in which the reaction was observed. The biowiki tables “organisms”, “organism_map”, “condition_list”, and “reaction_conditions” exist for this purpose, though these have not made their way into usage on the production version of the Wiki as yet.

II. SynBioSS Designer

Introduction to SynBioSS Designer

The SynBioSS Designer is a web-based component of the SynBioSS Suite (synbioss.sourceforge.net) which automatically generates a kinetic model from a construct composed entirely of BioBricks. It uses a simple set of biological rules to build the network of biomolecular interactions. The following is an overview of the algorithms used to generate sets of biomolecular reactions which describe BioBrick constructs as well as a tutorial for the use of the Designer.

Kinetic Model Generation

It is important to note that SynBioSS Designer operates as a biological rule-based algorithm. It does not conglomerate static lists of reactions associated with individual BioBricks; this is probably an impossible approach to a sophisticated problem. While the Designer does consider the behavior of individual parts, it also considers the spatial and temporal connectivity of the parts and biochemical interactions resulting from the system's myriad properties. As such, the Designer is context-sensitive, making it both accurate and versatile.

Importantly, the Designer generated reaction networks can be simulated using SynBioSS Desktop, a suite of reaction kinetics algorithms. The simulation results are probability distributions of all network components as a function of time. The user can then investigate the dynamic phenotypes generated by the synthetic biological system, investigate the influence of components and interactions on this phenotype and optimize the components and interactions to attain a targeted biological phenotype.

User Input Processing

A summary of the user's input is shown in the table below. The Designer references this information numerous times during the generation of kinetic models.

Table 2 – A summary of the information gathered by SynBioSS Designer through the web interface.

Input Type	**Properties**
BioBricks	Type (Promoter, RBS, Coding DNA, Terminator) Protein produced (if BioBrick is coding DNA) Location relative to other Bricks
Effectors	Protein to which it binds Maximum times which it may bind to said protein
Operators	Location (Upstream of -35, Between -35 and -10, Downstream of -10)
Promoters	Constitutively ON or OFF Associated operators
Proteins	Type (Activator, Reporter, Repressor) Number of subunits in an active complex Constitutive or Non-Constitutive Acts on its own or in concert with an effector (for Activators only)

Given the above information, the Designer begins by extracting all of the "transcriptional units" from the sequence of BioBricks. In the context of the Designer, a transcriptional unit is defined as a promoter followed by an RBS, coding DNA, and one or more terminators. Polycistronic mRNA is not yet supported. The GFP Producer described in the walkthrough contains two transcriptional units: R0040 to B0012, and R0062 to the second B0012. Recall that the former produces LuxR, while the latter produces GFP. The Designer thus associates R0040 regulation to LuxR production, R0062 regulation to GFP production, and stores this information internally for further reference.

Reaction Network Generation

Having extrapolated additional information from the user's input, the remainder of the Designer is dedicated to generating the kinetic model itself. The following sections address the types of reactions generated and the process by which they are generated. A few preliminary notes:

- Each time the Designer designates a reaction's stoichiometry, it simultaneously assigns each reaction 1) a rate law, based on the number of reactants or the nature of the reaction, and 2) a default kinetic constant of an appropriate order of magnitude, based on the nature of the reaction. By appropriate order we mean parameter values that are of the order of magnitude with similar reactions found in the tetracycline, lactose and arabinose operons. These are well studied operons and the reactions representing transcription, translation, regulation, and induction have been stored in the SynBioSS Wiki. We should stress that the Designer will generate the reaction network with example kinetic constant values, but the user should pay close attention to these parameters, should search the literature for accurate information pertinent to the system of his/her interest and examine the influence of the parameter values on the dynamic behavior.

- In the following sections, an ellipsis represents a sequence. For example, when Prot = LacI, Eff = IPTG, n = 4, and k = 4, the two following sets of reactions are equivalent:

1. $Prot_n + Eff \rightarrow Prot_n{:}Eff$
2. $Prot_n{:}Eff_1 + Eff \rightarrow Prot_n{:}Eff_2$

   …

k. $Prot_n{:}Eff_{(k-1)} + Eff \rightarrow Prot_n{:}Eff_k$

1. LacI4 + IPTG → LacI4:IPTG
2. LacI4:IPTG + IPTG → LacI4:IPTG2
3. LacI4:IPTG2 + IPTG → LacI4:IPTG3
4. LacI4:IPTG3 + IPTG → LacI4:IPTG4

Protein Multimerization

Regulatory proteins (Prot) often form complexes such as dimers (n = 2) or tetramers (n = 4) before binding to DNA. To generate these reversible multimerization reactions, the Designer loops through all user-input proteins and produces the following:

1. $2\ Prot \rightarrow Prot2$
2. $Prot2 \rightarrow 2\ Prot$

   …

2n-1. $2\ Prot_{(n/2)} \rightarrow Prot_n$

2n. $Prot_n \rightarrow Prot_{(n/2)}$

Basal Transcription

For each transcriptional unit which is constitutively ON, the Designer associates the promoter (Prom) with its operators (Op1, Op2, …, Op_n) and the protein whose expression it regulates (Prot) to generate appropriate transcription reactions. Transcription is represented by several steps, beginning with the reversible formation of the holoenzyme complex on the promoter:

1. $RNAp + Prom + Op1 + Op2 + \ldots + Op_n \rightarrow RNAp:Prom:Op1:Op2:\ldots:Op_n$
2. $RNAp:Prom:Op1:Op2:\ldots:Op_n \rightarrow RNAp + Prom + Op1 + Op2 + \ldots + Op_n$

Here we assume that the sequence of binding on DNA of the sigma factor and the RNA polymerase is not significant, and describe the holoenzyme with RNAp. Another important comment is that the reactions are still considered 2^{nd} order even though there are more than two reactants in the reaction. This is a valid assumption because the species cannot diffuse and react independently.

Closed-to-open conformational change follows:

3. RNAp:Prom:Op1:Op2:…:Op_n → RNAp:Prom:Op1:Op2:…:Op_n*

The next step is the escape of the holoenzyme complex from the promoter and involves the DNA which codes for the protein of interest:

4. RNAp:Prom:O1p:O2p:…:Op_n* → RNAp:DNA_Prot + Prom + Op1 + Op2 + … + Op_n

Finally, transcriptional elongation results in the mRNA corresponding to the protein of interest:

5. RNAp:DNA_Prot → RNAp + mRNA_Prot

Transcriptional elongation could in principle be modeled by a series of N first order reactions (N is the number of nucleotide bases in the DNA coding region), but since this would be computationally taxing and because the reaction times of each elongation step are exponentially distributed, transcriptional elongation is modeled with the reaction above in a gamma distributed process. This reaction has a kinetic constant and a number of steps associated with it.

Translation

Similarly to transcription, the Designer iterates through the transcriptional units, but instead produces reactions representing the generation of protein from mRNA. It is assumed that all genes are expressed one way or another, and thus even genes regulated by a promoter which is constitutively OFF are represented in the translation reactions. The first reaction represents the binding of the ribosome (rib) to the RBS on the mRNA transcript:

1. rib + mRNA_Prot → rib:mRNA_Prot

The second reaction describes the beginning of elongation and freeing of the RBS:

2. rib:mRNA_Prot → rib:mRNA_Prot_1 + mRNA_Prot

Finally, the protein itself is produced from the process of translational elongation:

3. rib:mRNA_Prot_1 → rib + Prot

Protein-DNA Binding

For every protein (Prot), the Designer loops through each operator (Op) with which the protein's active complex ($Prot_n$) may interact and generates the appropriate binding and unbinding reactions:

1. $Prot_n$ + Op → $Prot_n$:Op
2. $Prot_n$:Op → $Prot_n$ + Op

These reactions are not generated for proteins which do not bind to DNA, such as GFP, or for activator proteins which must act in concert with effector molecules, such as LuxR with HSL. Given that a protein complex does bind to DNA on its own, however, additional reactions are generated to describe the protein binding to non-specific sites (nsDNA):

3. $Prot_n$ + nsDNA → $Prot_n$:nsDNA
4. $Prot_n$:nsDNA → $Prot_n$ + nsDNA

Protein-Effector Binding

For every effector (Eff), the Designer considers the protein complex ($Prot_n$) to which it may bind as well as the maximum number of times which it may bind (k) to generate the following binding and unbinding reactions:

1. $Prot_n$ + Eff → $Prot_n$:Eff

2. $Prot_n:Eff \rightarrow Prot_n + Eff$

…

2k-1. $Prot_n:Eff_{(k-1)} + Eff \rightarrow Prot_n:Eff_k$

2k. $Prot_n:Eff_k \rightarrow Prot_n:Eff_{(k-1)} + Eff$

Protein-Effector-DNA Binding

Similar to Protein-Effector binding, except the Designer also iterates through the operators (Op) to which the protein complexes ($Prot_n$) can bind to. If the complex does not require an effector in order to bind to DNA, the reactions generated are:

1. $Prot_n:Op + Eff \rightarrow Prot_n:Eff:Op$

2. $Prot_n:Eff:Op \rightarrow Prot_n:Op + Eff$

…

2k-1. $Prot_n:Eff_{(k-1)}:Op + Eff \rightarrow Prot_n:Eff_k:Op$

2k. $Prot_n:Eff_k:Op \rightarrow Prot_n:Eff_{(k-1)}:Op + Eff$

2k+1. $Prot_n:Eff + Op \rightarrow Prot_n:Eff:Op$

2k+2. $Prot_n:Eff:Op \rightarrow Prot_n:Eff + Op$

…

4k-1. $Prot_n:Eff_k + Op \rightarrow Prot_n:Eff_k:Op$

4k. $Prot_n:Eff_k:Op \rightarrow Prot_n:Eff_k + Op$

If $Prot_n$ must be bound to an effector before it can bind to DNA (e.g. LuxR with HSL), then only the bottom two reactions are generated.

Aside from operators associated with individual promoters, the Designer generates Protein-Effector-DNA binding reactions with nsDNA as well. On such example is:

4k+1. $Prot_n:nsDNA + Eff \rightarrow Prot_n:Eff:nsDNA$

Activation/Leakiness

Activation and leakiness reactions have similar forms, in that they both involve slightly altered basal transcription reactions, with a protein complex bound to one operator instead of all operators being free. Activation and leakiness reactions are only generated for operators that are upstream of their associated promoter region. One interesting result of this is that the Designer recognizes steric repression. Any protein, even proteins that are nominally "activators", if bound to an operator site that is between the -35 and -10 regions or downstream of the -10 region will act as repressors due to the Protein-Operator and Protein-Effector-Operator reactions generated earlier.

To produce activation and leakiness reactions, the Designer iterates through all proteins, in turn iterating through all upstream operator sites to which these proteins bind. The reactions generated are shown below; in this case, the binding protein complex is denoted as "$Prot_n$" while the protein expressed by the transcriptional unit is called "$Prot_{exp}$":

1. $RNAp + Prom + Prot_n:Op1 + Op2 + \ldots + O_n \rightarrow RNAp:Prom:Prot_n:Op1:Op2:\ldots:Op_n$
2. $RNAp:Prom:Prot_n:Op1:Op2:\ldots:Op_n \rightarrow RNAp + Prom + Prot_n:Op1 + Op2 + \ldots + Op_n$
3. $RNAp:Prom:Prot_n:Op1:Op2:\ldots:O_n \rightarrow RNAp:Prom:Prot_n:Op1:Op2:\ldots:Op_n*$
4. $RNAp:Prom:Prot_n:Op1:Op2:\ldots:Op_n* \rightarrow RNAp:DNA\_Prot_{exp} + Prom + Prot_n:Op1 + Op2 + \ldots + Op_n$

If $Prot_n$ must work in concert with an effector in order to bind to DNA, the reactions are generated as follows:

5. $RNAp + Prom + Prot_n:Eff_k:Op1 + Op2 + \ldots + Op_n \rightarrow RNAp:Prom:Prot_n:Eff_k:Op1:Op2:\ldots:Op_n$

And so on.

Degradation

To represent degradation, the Designer loops through all proteins in the system and generates the following reaction:

1. Prot → Ø

The Designer may also generate additional degradation reactions according to a number of conditional statements. If the protein is not constitutively produced:

2. mRNA_Prot → Ø

If the protein binds DNA, but does not require an effector to do so:

3. $Prot_n$:nsDNA → nsDNA

If the protein binds to an effector:

4. $Prot_n$:Eff → Eff

…

k+4. $Prot_n$:Eff_k → k Eff

k+5. $Prot_n$:Eff:nsDNA → Eff + nsDNA

…

2k+5. $Prot_n$:Eff_k:nsDNA → k Eff + nsDNA

If the protein complex must act in concert with an effector, of the reactions listed immediately above, only the first sequence of reactions and very last reaction are generated.

Transport

Constitutively produced proteins are represented not as some initial amount of protein, but as a zero-order reaction for transport of the protein complex into the system:

1. $\emptyset \rightarrow Prot_n$

Default System Specifics

The Designer assigns initial amounts and a "split on cell division" property to all species. These can be altered once the gene network is uploaded in SynBioSS Desktop (DS). A summary of the default values is shown in Table 3 below.

Table 3 – Default initial conditions assigned by SynBioSS Designer.

Species Type	**Initial Amount**	**Split on Cell Division**
mRNA	0	Y
nsDNA	5000000	N
Promoters & Operators	1	N
Proteins	0	Y
Ribosome	600	N
RNAp	300	N
Other	0	N

Additionally, all species are designated to be saved to a file upon simulation, and the initial volume of the system is 1e-15L by default.

Tutorial

One example of a network which can be processed by the Designer is a "GFP Producer Controlled by 3OC6HSL Receiver Device" shown below:

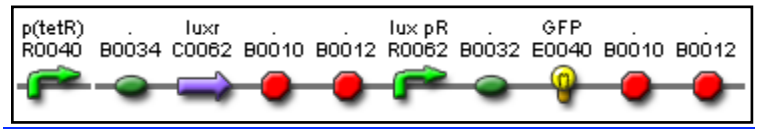


Figure 2 – A diagram of BioBrick T9002 as seen on the partsregistry.org website.

This composite part consists of two main regions. The first five Bricks describe a TetR repressible promoter (R0040) which regulates the expression of LuxR (C0062). The remaining five parts detail a promoter activated by LuxR in concert with HSL (R0062) which controls the production of GFP (E0040). Needed BioBrick ribosome binding sites (B0034 and B0032) and terminator sites (B0010 and B0012) are added in the DNA sequence.

In the presence of TetR, the function of the entire system is that of an AND gate for aTc (or tetracycline) and HSL. Without the effector HSL, LuxR cannot activate R0062 and GFP is not produced. Without the inducer aTc, TetR represses R0040 and no LuxR is produced, also inhibiting production of GFP.

All user input described in this tutorial can be found on the Parts Registry; no extensive research is required, although this may not be the case for poorly described BioBricks.

BioBrick Input and Specifics

The first step is to input the BioBricks in Designer. It is important to enter the BioBricks in order, from left to right, as shown in Figure 2. To input a brick in the web

interface, type in the brick's ID, select its "type" (e.g. promoter), and click the “Add BioBrick” button. Using a screenshot to visualize the step, the first brick is input as follows:

Figure 3 – The BioBricks input field for SynBioSS Designer

The next brick is “B0034”, type “RBS”, and so on, until all parts are added. Note that although parts C0062 and E0040 are represented by different symbols, they are both defined as “Coding DNA” in the SynBioSS Designer. After all of the parts are added, the "Current BioBricks" table should look like this:

CURRENT BIOBRICKS

BioBrick ID	Type	Specifics
R0040	Promoter	Constitutively: *required* Operators: *required*
B0034	RBS	
C0062	Coding DNA	*required*
B0010	Terminator	
B0012	Terminator	
R0062	Promoter	Constitutively: *required* Operators: *required*
B0032	RBS	
E0040	Coding DNA	*required*
B0010	Terminator	
B0012	Terminator	

Figure 4 – An example table depicting the bricks of Figure 2 entered into SynBioSS Designer

The Designer now needs additional information on coding DNA and promoters. For coding DNA, the user must enter the protein which is produced. To add this information, select the coding DNA's ID from the drop-down menu in the "Coding DNA Specifics" section, enter the protein's name, select the protein’s type, and click "Add Protein". In the case of C0062:

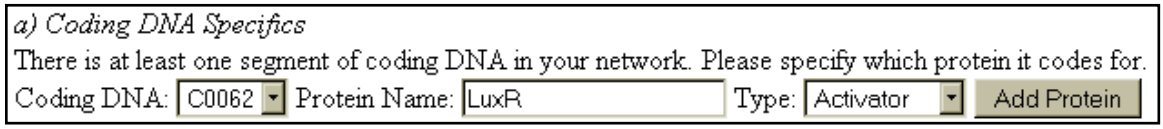


Figure 5 – The field for associating specific coding blocks with their protein products.

For promoters, the user must input 1) whether the promoter is constitutively ON or OFF, and 2) the operator sites associated with the promoter. To input the first piece of information, select the promoter's ID from the drop-down menu in the "Promoter Specifics" section and click either the "ON" or "OFF" button. According to the Parts Registry's page on R0040, this promoter is constitutively ON:

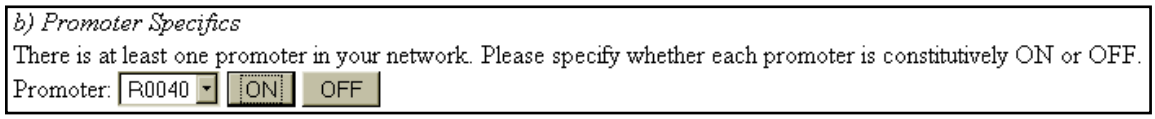


Figure 6 –This field is used to specify whether the promoter is normally active (and may be regulated through repression) or normally inactive (in which case it would require activation for gene transcription to occur).

The user must also input the name and location of the operator sites associated with each promoter. This information can also be found on each promoter's page, such as R0040. Figure 7 is a screenshot from the Registry of Biological Parts (partsregistry.org).

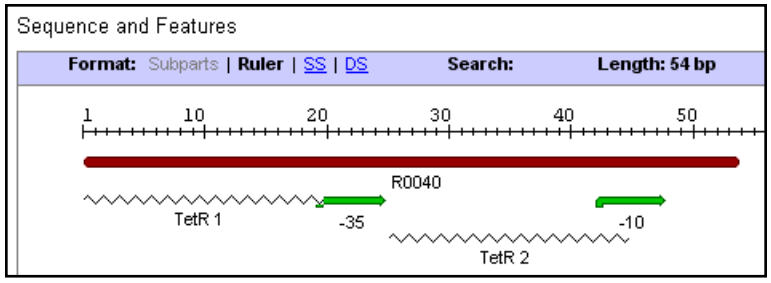


Figure 7 – This screenshot from the partsregistry.org website shows the location of the TetR sites within the R0040 promoter block.

This promoter contains two operator sites: one upstream of -35, and one between -35 and -10. To input operators, select the promoter's ID from the drop-down menu, enter a

name for the operator, select the operator's location relative to the promoter, and click "Add Operator". The name of the operator need not be the same as the one displayed on the Parts Registry; *all* operator sites must have unique names, however. We shall call the operators on R0040 "tetO1" and "tetO2" to prevent confusion with the TetR protein itself and its dimer. For the upstream operator:

Figure 8 – The user must specify the nature and position of any operator sites contained within the promoter blocks using this field.

These promoter specifics must also be entered for R0062. The page for R0062 claims that it "gives weak constitutive expression of downstream genes" but is also "pretty good off in the absence of LuxR/HSL". It is the user's choice whether to define R0062 as "OFF", or to define it as "ON" and adjust the final model's kinetic data appropriately. We shall define it as "OFF". We shall also name this promoter's operator site "luxO1".

Upon entering all coding DNA and promoter specifics, the "Current BioBricks" table is now complete:

CURRENT BIOBRICKS

BioBrick ID	Type	Specifics
R0040	Promoter	Constitutively: ON Operators: tetO1 (Upstream of -35) tetO2 (Between -35 and -10)
B0034	RBS	
C0062	Coding DNA	LuxR (Activator)
B0010	Terminator	
B0012	Terminator	
R0062	Promoter	Constitutively: OFF Operators: luxO1 (Upstream of -35)
B0032	RBS	
E0040	Coding DNA	GFP (Reporter)
B0010	Terminator	
B0012	Terminator	

Figure 9 – The complete table of BioBricks specified for the example system, along with all associated data.

Protein Input and Specifics

The next step is to specify additional information on the proteins in the system. If any constitutively expressed proteins affect the behavior of the system, they must be entered. In the case of our construct, we input TetR:

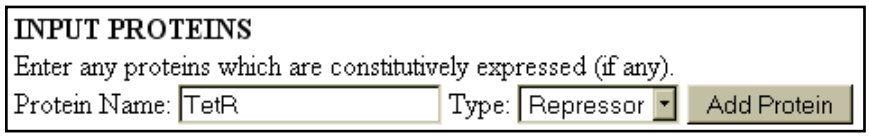


Figure 10 – The nature of all proteins mentioned in the system must be specified (reporter, repressor, or activator).

Next, we must input the number of protein subunits in an "active" complex of proteins. Some proteins, such as GFP, form no complexes of interest; thus the “active complex” is considered to have only one subunit. Other proteins may form dimers or larger complexes which bind to DNA. For example, two LuxR form LuxR2, a complex which interacts with luxO1 in concert with HSL. To input this data, select the protein from the drop-down menu in the "Complex Specifics" section, type in the number of subunits, and click "Add Complex Info". For LuxR:

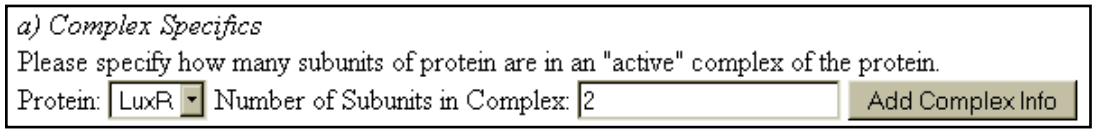


Figure 11 – Many proteins act as homodimers, homotetramers, etc., so the number of monomers per “active complex” must be specified.

We must also indicate the operator sites to which proteins may bind to as complex and/or in concert with effectors. To do so, select the protein and one operator which it binds from the drop-down menus in the "Binding Specifics" section, and click "Add Binding

Info". The order in which one enters the operators does not matter, and not all proteins need to bind to something. For LuxR:

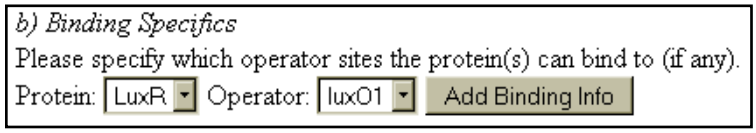


Figure 12 – It is important to specify which proteins (among the repressors and activators) bind to which operator sites. This is performed in the following field.

Effector Input and Specifics

The final step is to specify information on the effector (inducer) molecules in the system, if any. Recall that the inducer aTc and the effector HSL are important species in our GFP generator system. Adding aTc:

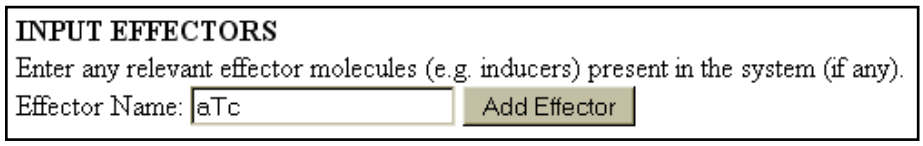


Figure 13 – Any small molecules which influence repressor or activator protein must be entered here.

Finally, we must specify how many times each effector can bind to a protein complex. In our example system, aTc can bind to TetR2 a maximum of two times, and HSL can bind to LuxR2 a maximum of two times. To input this information, select the effector and protein from the drop-down menus in the "Effector Specifics" section, enter the maximum number of times the inducer can bind to the complex (*not* the individual protein, unless it does not form a complex), and click "Add Inducer Info". For aTc:

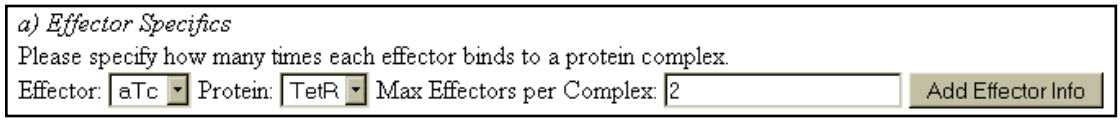
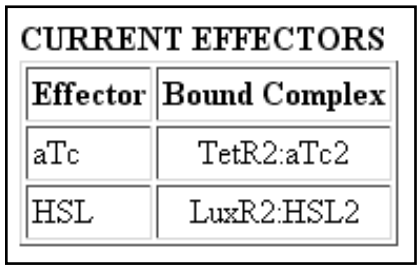


Figure 14 – Many active complexes of protein can bind more than one small molecule. The maximum number of bound molecules is entered here.

Repeating this input for HSL, the "Current Inducer" table is now finished:



CURRENT EFFECTORS

Effector	Bound Complex
aTc	TetR2:aTc2
HSL	LuxR2:HSL2

Figure 15 – This is what the completed effecter table should look like after the preceding information has been entered.

The Designer now has enough information to generate a detailed, molecular-level model for this system of BioBricks.

Example Output

The Designer-generated reaction network for the GFP Producer construct is detailed in the following table. Note several important general and system-specific biological behaviors which the Designer recognizes and represents accurately:

- Activators and repressors only bind to DNA as active multimers
- Promoters are associated with the correct coding regions
- Basal transcription only occurs for the promoter which is constitutively ON
- Effectors bind to proteins an appropriate number of times
- LuxR only binds to DNA in concert with HSL (LuxR2:HSL2)
- Reporters do not bind to non-specific DNA regions
- Activation and leakiness only occur on operators upstream of their promoter
- Activation and leakiness are differentiated by a kinetic constant
- Only non-constitutively produced proteins have associated mRNA species
- Constitutively expressed proteins are imported into the system
- Transcriptional and translational elongation events are gamma-distributed

Unless otherwise noted, all reactions have elementary rates laws with macroscopic kinetic constants in terms of moles, liters, and seconds.

Table 4 – An example of the reaction network generated by SynBioSS Designer for the example network. This network would be exported as a NetCDF or SBML file.

Protein Multimerization	**Kinetic Data**
2 TetR --> TetR2	1000000000
TetR2 --> 2 TetR	10
2 LuxR --> LuxR2	1000000000
LuxR2 --> 2 LuxR	10
Transcription	
RNAp + R0040 + tetO2 + tetO1 --> RNAp:R0040:tetO2:tetO1	10000000
RNAp:R0040:tetO2:tetO1 --> RNAp + R0040 + tetO2 + tetO1	0.75
RNAp:R0040:tetO2:tetO1 --> RNAp:R0040:tetO2:tetO1*	0.3
RNAp:R0040:tetO2:tetO1* --> RNAp:DNA_LuxR + R0040 + tetO2 + tetO1	30 nt/s
RNAp:DNA_LuxR --> RNAp + mRNA_LuxR	30 nt/s, 660 nt
Translation	

rib + mRNA_LuxR --> rib:mRNA_LuxR	100000
rib:mRNA_LuxR --> rib:mRNA_LuxR_1 + mRNA_LuxR	33 aa/s
rib:mRNA_LuxR_1 --> rib + LuxR	33 aa/s, 220 aa
rib + mRNA_GFP --> rib:mRNA_GFP	100000
rib:mRNA_GFP --> rib:mRNA_GFP_1 + mRNA_GFP	33 aa/s
rib:mRNA_GFP_1 --> rib + GFP	33 aa/s, 220 aa
Regulation	
TetR2 + tetO1 --> TetR2:tetO1	1000000000
TetR2:tetO1 --> TetR2 + tetO1	0.005
TetR2 + tetO2 --> TetR2:tetO2	1000000000
TetR2:tetO2 --> TetR2 tetO2	0.005
LuxR2 + HSL --> LuxR2:HSL	50000000
LuxR2:HSL --> LuxR2 + HSL	0.1
LuxR2:HSL + HSL --> LuxR2:HSL2	50000000
LuxR2:HSL2 --> LuxR2:HSL HSL	0.1
LuxR2:HSL2 + luxO1 --> LuxR2:HSL2:luxO1	1000000
LuxR2:HSL2:luxO1 --> LuxR2:HSL2 + luxO1	0.4
Induction	
TetR2 + aTc --> TetR2:aTc	50000000
TetR2:aTc --> TetR2 + aTc	0.1
TetR2:aTc + aTc --> TetR2:aTc2	50000000
TetR2:aTc2 --> TetR2:aTc + aTc	0.1
TetR2:aTc + tetO1 --> TetR2:aTc:tetO1	1000000000
TetR2:aTc:tetO1 --> TetR2:aTc + tetO1	0.7
TetR2:tetO1 + aTc --> TetR2:aTc:tetO1	1000000
TetR2:aTc:tetO1 --> TetR2:tetO1 + aTc	0.4
TetR2:aTc2 + tetO1 --> TetR2:aTc2:tetO1	1000000
TetR2:aTc2:tetO1 --> TetR2:aTc2 + tetO1	0.4
TetR2:aTc:tetO1 + aTc --> TetR2:aTc2:tetO1	50000000
TetR2:aTc2:tetO1 --> TetR2:aTc:tetO1 + aTc	0.1
TetR2:aTc + tetO2 --> TetR2:aTc:tetO2	1000000000
TetR2:aTc:tetO2 --> TetR2:aTc + tetO2	0.7
TetR2:tetO2 + aTc --> TetR2:aTc:tetO2	1000000
TetR2:aTc:tetO2 --> TetR2:tetO2 + aTc	0.4
TetR2:aTc2 + tetO2 --> TetR2:aTc2:tetO2	1000000
TetR2:aTc2:tetO2 --> TetR2:aTc2 + tetO2	0.4
TetR2:aTc:tetO2 + aTc --> TetR2:aTc2:tetO2	50000000
TetR2:aTc2:tetO2 --> TetR2:aTc:tetO2 + aTc	0.1
Non-Specific DNA Interactions	
TetR2 + nsDNA --> TetR2:nsDNA	1000
TetR2:nsDNA --> TetR2 + nsDNA	1.6225
TetR2:aTc + nsDNA --> TetR2:aTc:nsDNA	1000000000
TetR2:aTc:nsDNA --> TetR2:aTc + nsDNA	0.7
TetR2:nsDNA + aTc --> TetR2:aTc:nsDNA	1000000
TetR2:aTc:nsDNA --> TetR2:nsDNA + aTc	0.4
TetR2:aTc2 + nsDNA --> TetR2:aTc2:nsDNA	1000000
TetR2:aTc2:nsDNA --> TetR2:aTc2 + nsDNA	0.4

TetR2:aTc:nsDNA + aTc --> TetR2:aTc2:nsDNA	50000000
TetR2:aTc2:nsDNA --> TetR2:aTc:nsDNA + aTc	0.1
LuxR2:HSL2 + nsDNA --> LuxR2:HSL2:nsDNA	1000000
LuxR2:HSL2:nsDNA --> LuxR2:HSL2 + nsDNA	0.4
Activation	
RNAp + R0062 + LuxR2:HSL2:luxO1 --> RNAp:R0062:luxO1:LuxR2:HSL2	10000000
RNAp:R0062:luxO1:LuxR2:HSL2 --> RNAp + R0062 + LuxR2:HSL2:luxO1	0.75
RNAp:R0062:luxO1:LuxR2:HSL2 --> RNAp:R0062:luxO1:LuxR2:HSL2*	3
RNAp:R0062:luxO1:LuxR2:HSL2* --> RNAp:DNA_GFP + R0062 + LuxR2:HSL2:luxO1	30
Leakiness	
RNAp + R0040 + TetR2:tetO1 + tetO2 --> RNAp:R0040:tetO2:tetO1:TetR2	10000000
RNAp:R0040:tetO2:tetO1:TetR2 --> RNAp + R0040 + TetR2:tetO1 + tetO2	0.75
RNAp:R0040:tetO2:tetO1:TetR2 --> RNAp:R0040:tetO2:tetO1:TetR2*	0.3
RNAp:R0040:tetO2:tetO1:TetR2* --> RNAp:DNA_LuxR + R0040 + TetR2:tetO1 + tetO2	30
Transport	
--> TetR2	1.00E-10
Degradation	
GFP -->	0.000778
mRNA_GFP -->	0.0015
TetR -->	0.000778
TetR2:nsDNA --> nsDNA	0.000193
LuxR -->	0.000778
mRNA_LuxR -->	0.0015
TetR2:aTc --> aTc	0.000289
TetR2:aTc:nsDNA --> aTc + nsDNA	0.000193
TetR2:aTc2 --> 2 aTc	0.000289
TetR2:aTc2:nsDNA --> 2 aTc + nsDNA	0.000193
LuxR2:HSL --> HSL	0.000289
LuxR2:HSL2 --> 2 HSL	0.000289
LuxR2:HSL2:nsDNA --> 2HSL + nsDNA	0.000193

Again, please note that for some reactions (e.g. aTc binding on TetR, or TetR binding on tetO1) there are experimental measurements providing the kinetic constant values. These are certainly context dependent and the user should investigate the accuracy of the parameters and their applicability. Designer will provide default values for the kinetic constants as a matter of necessity, but these are merely default values necessary for populating the reaction network skeleton. A more thorough examination of the correct values based on the literature (or SynBioSS Wiki) is still required.

III. SynBioSS Desktop Simulator

The SynBioSS simulator is described and referenced in the main text more thoroughly than the Wiki or Designer, so the materials presented in the supplement will be brief.

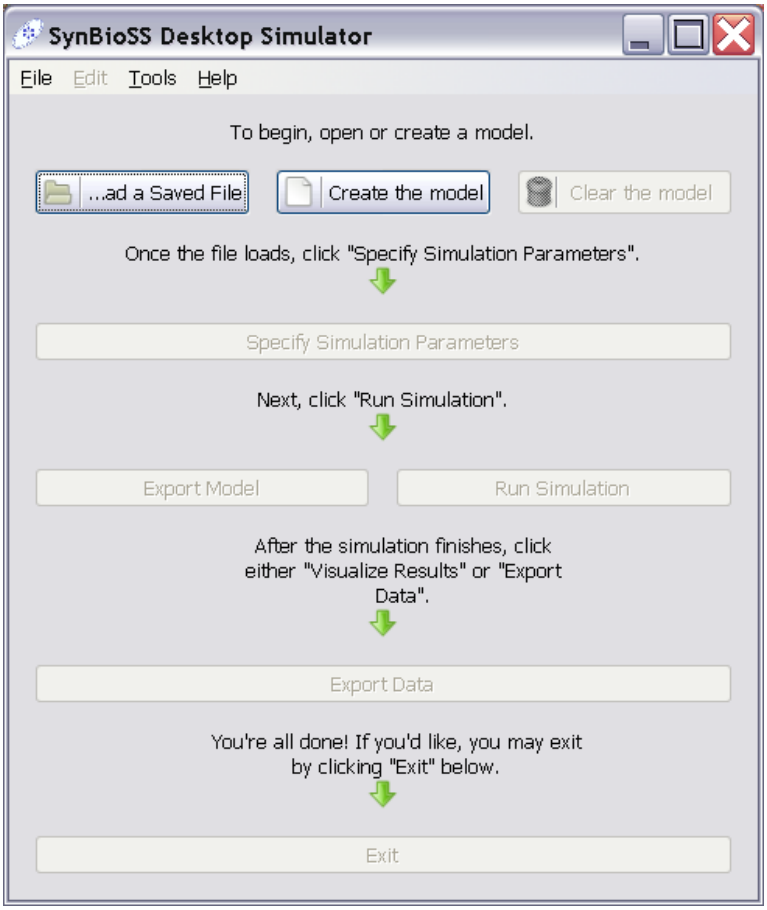


Figure 16 – The main window of SynBioSS Desktop as it appears immediately after opening the application. As each "level" of the simulation is completed, starting by opening an existing model or defining a new one, the appropriate options on the next level will become available.

Figure 16 depicts the main window of the SynBioSS Desktop as it appears immediately after launching the application. This window is the primary means by which users interact with the application. A new model can be opened (in SBML or NetCDF format) through the "Read a Saved File" button at top-left, or a new model can be created

using the “Create the model” button at top-center. In either case, the model can be edited, and the edit/create window is shown in Figure 17.

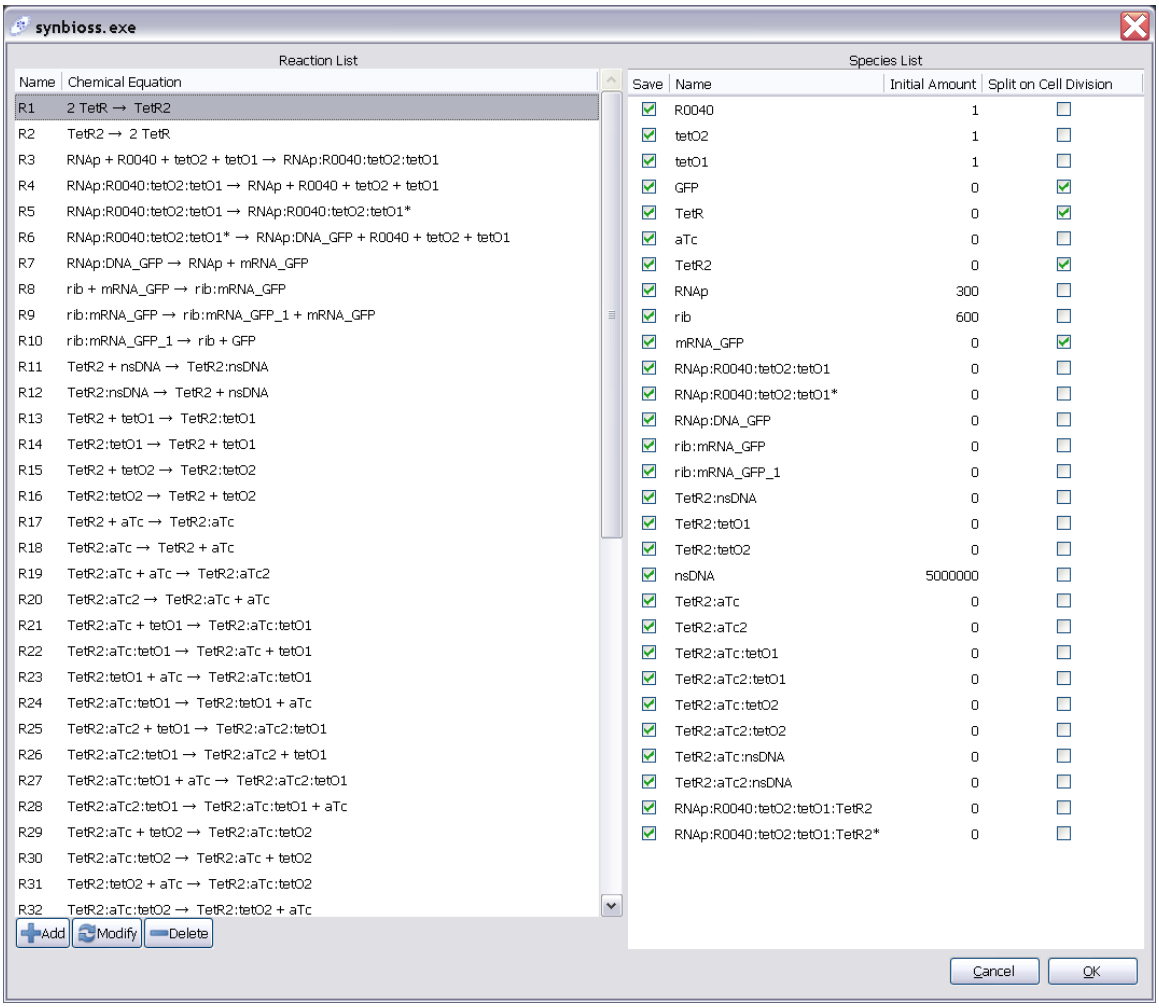


Figure 17 – The network creation/editing window of SynBioSS Desktop Simulator.

After the model (reactions stoichiometry, kinetics, and initial conditions) has been specified, the second level option on the main window, “Specify Simulation Parameters,” becomes active. Users must then click on this button to interact with the window shown in Figure 18, specifying the length of the simulation, the number of trajectories to simulate, and other important parameters.

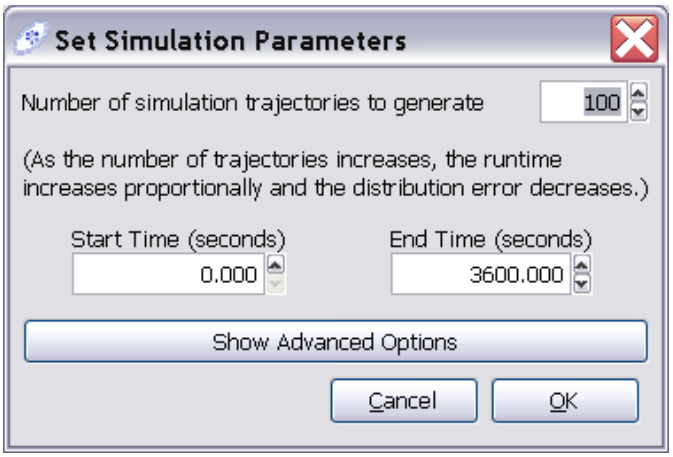


Figure 18 – The dialog box used to specify simulation parameters other than the model itself.

After the model has been defined and the auxiliary parameters specified, the third level of options of Figure 16 become available. At this point, users may export the model to a new NetCDF or SBML file, or they may run the simulation. After the simulation has been run, the data may be exported to a set of .csv files (comma separated values, one file per species) for analysis in any plotting package of the user's choice.

References

1. "MediaWiki database tables." MediaWiki. 24 July 2004. MediaWiki Foundation. 6 Aug 2007 <http://www.mediawiki.org/wiki/Category:MediaWiki_database_tables>.

2. "LocalSettings.php." MediaWiki. 30 July 2007. MediaWiki Foundation. 6 Aug 2007 <http://www.mediawiki.org/wiki/Localsettings>.