Frameworks for Shareable Multiscale Modeling

Goals, Philosophy, Tools, Languages and Needs

James P. Sluka, James A. Glazier, and Maciej Swat

Biocomplexity Institute
Indiana University

Scope

Multiscale biological modeling (MSM) is at a turning point in its development. As models become more complex, more robust, and more useful it is becoming clear that there will not be a “one size fits all” approach. Multiple models, instantiated in multiple computational modalities and spanning multiple scales will most likely be the long term paradigm in MSM. This heterogeneous approach presents unique problems in model definition, validation, storage, mining and reuse. As MSM moves from the basic research domain into the regulatory and medical domains it will become increasingly important that models are documented, transparent and reproducible. We believe these requirements require a robust, defensible, logical and publishable method of describing all phases of biological model development. In the short term, a suitable descriptive language, and the tools needed to use that language, will significantly help the MSM community develop “best practice” standards. In the long term those standards will be the basis for the much more exacting requirements that the longer term medical and regulatory applications will require.

The goals of this whitepaper are: 1) To present key concepts of multiscale biological modeling. 2) To present a language hierarchy and tool Framework for (1) that clarifies the work-flow for multiscale model and simulation development. 3) To describe how (2) supports model sharing, integration and model transport among modeling and simulation methodologies. 4) To identify key enabling components missing from current model representations, especially with reference to multiscale, multi-cell models and simulations. 5) To enable the treatment of repositories of models and results as mineable data.

This whitepaper will specifically **not** discuss particular simulation methodologies or applied-mathematics techniques for parameter propagation, coarse graining, etc. It will argue that the use and support of intermediate layers of model abstraction is essential to the goals of MSM, model sharing, mining, validation and reuse.

Framework Goals

This whitepaper will discuss the requirements and philosophy for a Model Development Framework that will enable the efficient generation, reuse and support of complex multiscale biological models and associated simulations. We will generally assume that these models include the multi-cell level, *i.e.* that they represent many individual cells or small cell clusters explicitly in space, roughly corresponding to what is seen through a 20 × microscope objective. However, the generic concepts and goals apply to a much broader spectrum of modeling scales.

To support multiscale model development, integration and curation, any Integration Framework should support the following requirements:

1) **Models as searchable data**: Models should exist in representations that allow searching using standard web tools. This requires tools to facilitate the creation, curation and use of repositories of models.

2) **Experimental data as model:** The Framework should allow annotated experimental data to be treated as a model.
3) **Future proofing using multiple layers of abstraction:** Because both computational resources, solver methods and the applied mathematics used to describe specific mathematical approaches are constantly evolving, the Framework must provide primary levels of model description abstraction which provide the ability to specify models (at any spatial level) in a way that insulates the model description of the underlying biology from any data related to the specific **methodology** used to solve the corresponding simulation. *E.g.* a model describing a set of genetic, regulatory or metabolic pathways should be solvable using ODEs, Gillespie or other methodologies; a model of tissue development should be solvable using GGH, Finite Element, Center-Model or Vertex Model methodologies; switching from single processor to multi-core, GPU or cluster computation.
4) **Plugability:** The Framework should allow users to combine multiple computational models at the same or different spatial scales to create larger models without excessive demands for user input. This ability requires the creation of repositories of reusable components which can be modified and extended. *E.g.* the assembly of a signaling pathway model with a cell-cycle regulation pathway model; the assembly of a cell-cycle pathway model with a model of tumor growth at the multi-cell scale and a tissue-scale model of nutrient supply.
5) **Encapsulation:** The Framework should allow the user to package any combination of models into an encapsulated model, *e.g.* the definition of a set of pathways models and cell behavior models into a model of a cell type; or the assembly of multiple cell types into a model of a tissue or organ.
6) **Templating and Reusability:** The Framework should allow the user to replace model or submodel parameters without changing the structure of a model, *e.g.* replacing the parameters in a human hepatocyte cell model with those appropriate for a rat to create a rat hepatocyte model.
7) **Refinement:** Any parameter or coarse-grained concept should be replaceable with a model generating that parameter or concept from data at a finer level of detail. Refinement and Encapsulation are essential to create multiscale models which can serve as ‘adjustable zoom’ microscopes.
8) **Consistent treatment of experimental and simulation data:** The Framework should use identical approaches to describe simulation and experimental data to allow the user to annotate both simulation output and experimental data so they can be compared quantitatively and qualitatively.
9) **Traceability:** The Framework should allow, and preferably automatically, annotate the model to facilitate data archiving, searching, mining and versioning.
10) **Naturalness:** The descriptors used at each modeling level should, as much as possible, correspond to the natural concepts at that level. *I.e.* the language for specifying Biological Models should correspond closely to standard biological concepts, while languages used for specifying Computational Models should correspond to software engineering concepts. The Framework should make these transitions of concept grouping automatic when translating between levels of abstraction.

11) **Simplification:** The Framework should encourage users to develop models at the highest possible level of abstraction to encourage reusability, mineability and Future proofing.

Together these requirements will greatly facilitate the broader aims of the MSM Framework and the absence of any of them is likely to seriously impede these goals. These goals represent the development of a **Framework** that will greatly facilitate model definition, sharing, validation and reuse at both single and multiple scales.

Definitions

Terminology in the area of biological modeling is abstruse, inconsistent and confusing. A few definitions of how we will use important terms may help:

Ontologies are lists of terms, their definitions and relationships. They include both **non-instantiable** categorical ontologies (like the Foundational Model of Anatomy FMA (Rosse 2003) and the Gene Ontology GO (Ashburner 2000)) and **instantiable** ontologies that we use to describe a model or part of a model (like Ontology of Physics for Biology OPB (Cook 2008) or a markup language).

Languages include both mark-up type descriptions (which have a lot in common with ontologies as noted above) and the “native” Languages of particular simulation environments. In this document, “Language” does not generally refer to a particular programing language, such as C++ or Python, but instead refers to a suite of computational objects that represent the concepts to be modeled or the capabilities of a particular simulation environment.

Tools are programs or subprograms that allow manipulation of models and language components and/or instantiate their concepts. Tools align with the appropriate Ontologies at particular levels of abstraction.

(Computational) Modality refers to the details of a particular computational instantiation. For example, a model implemented as a state machine versus a system of ODEs.

Use Cases are “proof of concept” models. They describe the domain to which a set of Languages, Tools and modalities apply. Use Cases specify the scope and types of questions the simulation environment can answer.

A **Toolkit** is an ensemble of ontologies, languages and tools relevant to a modeling and simulation domain.

A **Framework** is an ensemble of ontologies, languages and tools (a Toolkit) plus appropriate Use Cases relevant to a modeling and simulation domain.

Multi-cell models and simulations span the size domain from single cells to tissues. A Multi-cell model can have varying levels of sub-cellular detail; the exact level of detail depends on the behaviors that the individual cells (or set of cells) must exhibit.

Model is a generic term which can apply to nearly any concept or observation in science. It may refer to a computational model, to an experimentally observable property or behavior, or to a conceptual model that rationalizes an observable process or property.

A **Submodel** is a model used as a component of a larger or more comprehensive model.

A **Biological Model** is a formal description of a biological process in a human readable controlled language based on an Ontology. The Biological Model description stops where the qualitative description starts to becomes quantitative**.**

A **Quantitative Model** (we lack a natural term for this level of modeling) is a model which provides a minimal unambiguous template for quantitative specification but may lack specific parameter values. The natural bases for such models are "Interchange Languages".

A **Computational Model** refers to the mathematical form(s) and classes of algorithms used to describe a particular process or interaction. A Computational Model is a calculable, at least theoretically, representation of a Quantitative model, which in turn is a representation of a Biological Model. The natural bases for such Computational Models are APIs (advanced programming interfaces).

A **Simulation** is the specific code that implements a particular Computational Model. A Simulation is a computable representation of a Computational Model. The natural basis for such models are programming languages (Python, C++, Matlab, Mathematica …).

An **Interchange Language(s)** provide the mapping between the ontological descriptions at various levels in the model hierarchy.

APIs (Application Programming Interfaces) describe the input requirements and the output format of computational code designed to carry out various computational steps within a particular modality.

Introduction

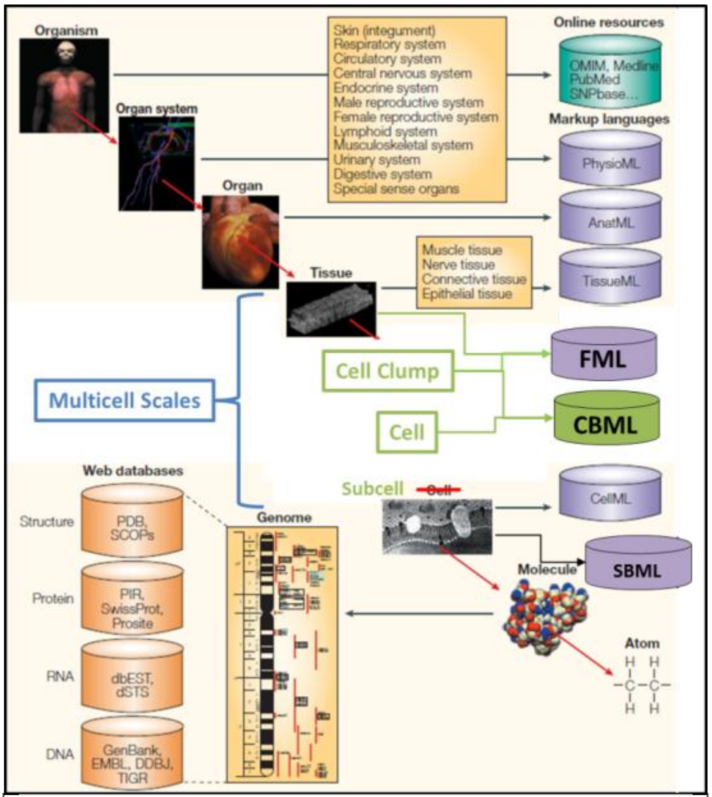


Figure 1: Biological scales and the corresponding computational tools and ontologies. (After P. Hunter & T. Borg, Nature Reviews Molecular Cell Biology 4, 237-243 (2003).) Elements added to show the missing scales are shown in green and blue.

Developmental biology, tissue engineering, regenerative biology, cancer biology and other biological fields are becoming quantitative experimental sciences. In many cases such research shares a focus on multi-cellular phenomena (*i.e.*, associations of cells, ECM and organ systems). As time-lapse imaging of 3D protein localization at cell and subcell level becomes more common, researchers face a new type of data deluge. This new flood of data is beginning to provide foundational information on the behavior of cells and tissues. Biologists will need to develop tools to describe, understand and control the processes underlying their observations. The increasing quantitative content of experiments has enabled the development of reliable computational models of biological and chemical processes at selected spatial and temporal scales (see Figure 1). Figure 1 adapts the Physiome framework and specifies many of the components needed for a useable infrastructure to support multiscale biological modeling. However, the Physiome infrastructure is incomplete. In Figure 1 we have added two biological scales (cell and "cell clump") and some of the infrastructure, in this case and proposed markup language "CBML", needed to complete the range of biological scales from the molecular scale to the organism scale.[1] The Physiome framework specifies many of the components needed for a useable infrastructure to support multiscale biological modeling but is not complete. To fully realize MSM additional infrastructure will need to be added to the Physiome framework.

As computational biology tools have evolved it has created a growing need to be able to integrate models at a larger scale with finer models operating at lower scales (Figure 2). Improving techniques for simulating subcell (molecular), cell, multi-cell and organ-level processes and combining those models into a robust and scientifically useful tool require concerted effort by many researchers. Large-scale simulation of complex, multiscale biological systems integrates a broad span of investigator expertise

[1] Because of its history, Physiome lacks a concept of a *cell* as a spatially extended motile (active) agent. Despite its name, CellML actually describes molecular-level chemical and electrical objects and their behaviors. CBML and CBO are respectively, a proposed interchange language and ontology to represent the missing "cells as agents" scale.

ranging from medicine to biology to computational biology to computer and knowledge sciences. Because of the lack of a common (unified) framework current practice requires an early commitment to particular modeling modalities and the generation of platform-dependent code bases. *Locating, reusing, recombining, and adapting legacy models require arduous hand-coding methods and considerable expertise in multiple modeling platforms.*

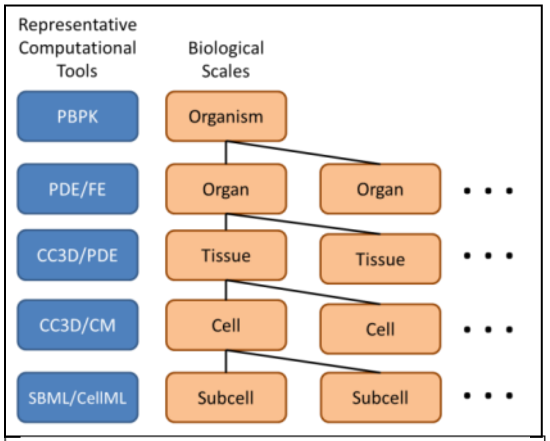


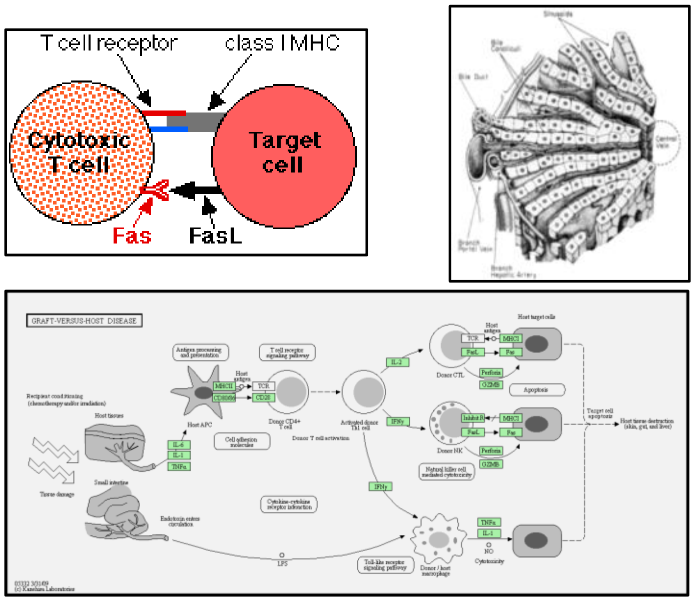
Figure 2: *Representative* Computational methodologies for various biological scales.

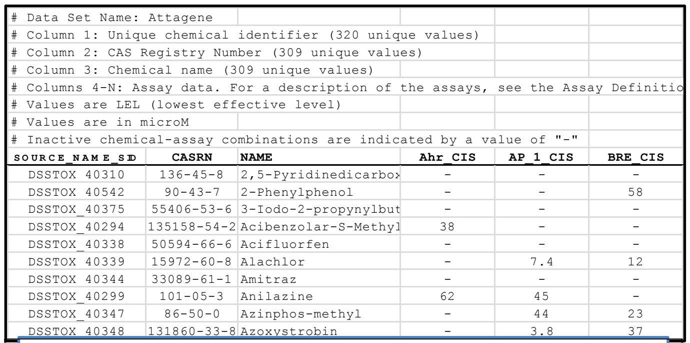
What is a Model?

At some level, all of science is "models". In the biological domain, models range from simplified "blob" diagrams that might be found in a freshman biology textbook, to complex interaction maps such as a KEGG pathway, to more complex representations such as the electro physiochemical models of the heart (Figure 3). Importantly, <u>biological experiments and experimental results are also models</u>. Indeed it is the wet-lab "models" upon which most computational models are not only based upon but also validated against.

In order to *effectively* describe a computa-tional biology model it must be possible to describe it first in purely biological terms. This should be the first step in the development of a computational biological model. At the <u>biological level</u> the model (or module) is guaranteed to be "mineable", shareable and "hot-swappable" with other models (or modules). In addition, it is the biological description that defines how models at different spatial (or temporal) scales interact.

People often use the terms "model" and "simulation" interchangeably. We will refer to specific code executable by a **simulation environment** and its internal representation during execution as **simulations**.





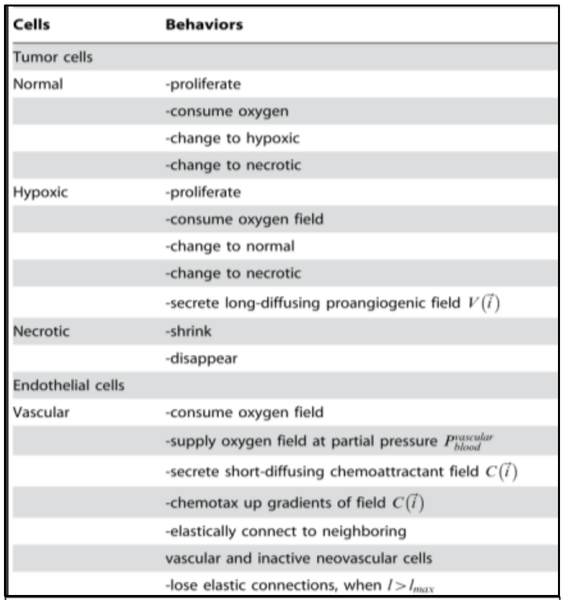
```
# Data Set Name: Attagene
# Column 1: Unique chemical identifier (320 unique values)
# Column 2: CAS Registry Number (309 unique values)
# Column 3: Chemical name (309 unique values)
# Columns 4-N: Assay data. For a description of the assays, see the Assay Definitio
# Values are LEL (lowest effective level)
# Values are in microM
# Inactive chemical-assay combinations are indicated by a value of "-"
```

SOURCE_NAME_SID	CASRN	NAME	Ahr_CIS	AP_1_CIS	BRE_CIS
DSSTOX 40310	136-45-8	2,5-Pyridinedicarbox	-	-	-
DSSTOX 40542	90-43-7	2-Phenylphenol	-	-	58
DSSTOX_40375	55406-53-6	3-Iodo-2-propynylbut	-	-	-
DSSTOX_40294	135158-54-2	Acibenzolar-S-Methyl	38	-	-
DSSTOX_40338	50594-66-6	Acifluorfen	-	-	-
DSSTOX 40339	15972-60-8	Alachlor	-	7.4	12
DSSTOX 40344	33089-61-1	Amitraz	-	-	-
DSSTOX_40299	101-05-3	Anilazine	62	45	-
DSSTOX_40347	86-50-0	Azinphos-methyl	-	44	23
DSSTOX 40348	131860-33-8	Azoxystrobin	-	3.8	37

Figure 3: Biological Models; "Blob" (top left), tissue cartoon (top right), KEGG pathway (center) and high throughput screen (bottom).

Current State of Model Description

Before we can <u>effectively</u> integrate multi-scale models, or indeed before we can adequately describe a single-scale model, we must have an agreed upon framework with which to describe the <u>underlying biology</u>. Without a unified biological description technology, the goal of multi-scale modeling, model (and module) sharing and reuse will be greatly inhibited.



Cells	Behaviors
Tumor cells	
Normal	-proliferate
	-consume oxygen
	-change to hypoxic
	-change to necrotic
Hypoxic	-proliferate
	-consume oxygen field
	-change to normal
	-change to necrotic
	-secrete long-diffusing proangiogenic field $V(\vec{i})$
Necrotic	-shrink
	-disappear
Endothelial cells	
Vascular	-consume oxygen field
	-supply oxygen field at partial pressure $P_{blood}^{vascular}$
	-secrete short-diffusing chemoattractant field $C(\vec{i})$
	-chemotax up gradients of field $C(\vec{i})$
	-elastically connect to neighboring
	vascular and inactive neovascular cells
	-lose elastic connections, when $l > l_{max}$

Figure 4: Representative *ad hoc* ontological description of a computational biology model. Only about one half of the full table is shown here.

It seems to us that what is first needed is a definition of the modular components at the biological level. This "Biological Description" is agnostic to the details of any particular computational framework. The Biological Description is both the first level of model description and is also the first level of model validation. The Biological Description should be in the language of modern biology and may contain little, or even no, mathematical details, let alone details of a particular computational instantiation.

The Biological Model description level is the natural level for model searching, sharing and reuse and for linking of models across multiple scales. Since the Biological Model is agnostic to computational details the details of implementation of a particular module are less of an obstacle to the task of combining (or reusing) models.

We believe that this Biological Model, in a form that a wet-lab biologist would be comfortable with, is a required precondition to <u>effectively</u> developing interoperable multiscale models.

Computational biologists already often use an *ad hoc* ontological description of their models. Computational biology publications often include a table similar to Figure 4 (Shirinifard 2009). The table in Figure 4 represents an "ontological description" of the particular model but it was created without an actual ontology (since a suitable ontology does not exist). The *ad hoc* nature of this common publication technique makes locating, interpreting, validating and reusing the model extremely difficult.

The *ad hoc* nature of the "ontological description" also tends to obscure the choices that were made in translating the Biological Model into a Computational Model. Details, such as the representation of space and time in the computational instantiation, are not included in the "ontological description" but instead must be culled from the text of the paper or worse yet, from references within references in the paper. For example, the ad hoc description in Figure 4 lacks a definition of the time step used in the ODE solvers and of the spatial scale of the biological model represented by the actual computational model. We have then a situation where the complete workflow from biological observation to computational model is done, often without explicit recognition of the process, as shown in figure 5.

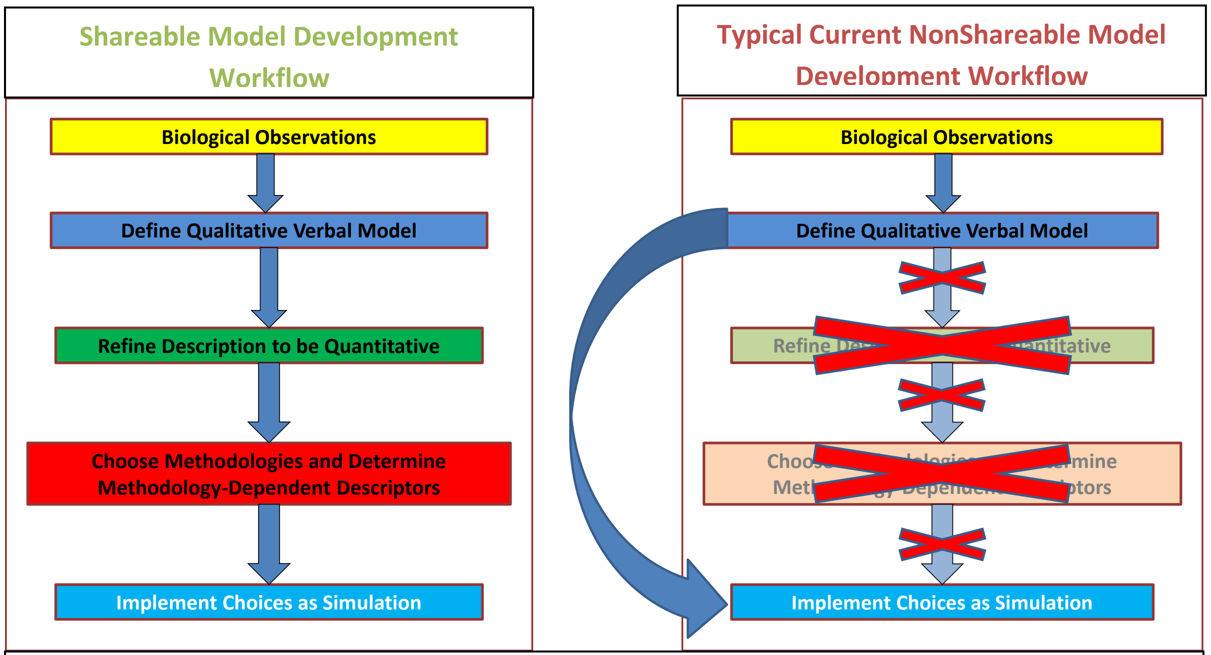

Shareable Model Development Workflow
Biological Observations
Define Qualitative Verbal Model
Refine Description to be Quantitative
Choose Methodologies and Determine Methodology-Dependent Descriptors
Implement Choices as Simulation
Typical Current NonShareable Model Development Workflow
Biological Observations
Define Qualitative Verbal Model
Implement Choices as Simulation

Figure 5: Shareable (Left) and Non-shareable (Right) Model Development Workflows. Most current modeling environments encourage non-shareable model development and most model developers follow a non-shareable workflow.

This work flow obscures choices that were made and does not capture the model at intermediate levels of definition such as the biological and mathematical. Indeed, the critical starting level of the model definition, the description of the biology being modeled, is often lost.

Therefore we believe that before any significant progress can be made in multi-scale modeling and model (module) sharing, reuse and validation there must first be a useable formal ontology that can, at least, describe the observed biology and the types of models that a wet-lab biologist is familiar with. This formal ontology should be completely agnostic to both the mathematical description of the system and to the computational framework(s) in which it might be instantiated.

Shareable Model Definition Workflow

To develop a simulation of a biological phenomenon in a shareable fashion, we begin with a qualitative verbal description; a "Biological Model" based on biological data, described using suitable reference ontologies, and eventually transform it into a script that a simulation environment can execute (Figure 6). At the top, we start with biological data and a biological model, as it might be described by a wet-lab biologist, or biology textbook. This level might include genes, proteins, behaviors, cells, tissues and interactions. Existing ontologies (such as FMA and GO) act as "naming authorities" that link objects and concepts to

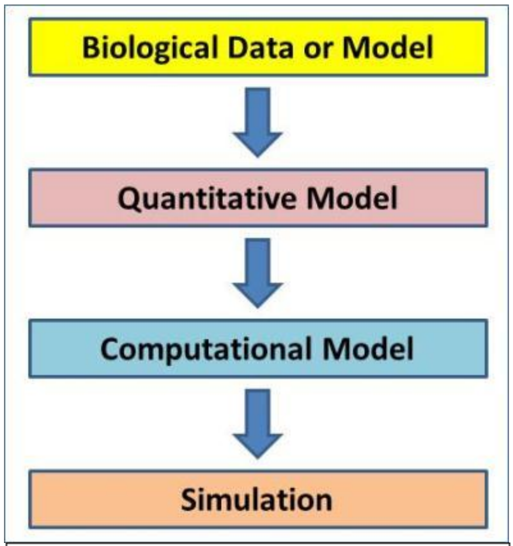


Figure 6: A well-defined workflow for computational biology model development.

other resources. The ontologies guarantee the “mineability” of the model descriptions and should provide linkages to existing descriptions of concepts and entities included in the particular process being modeled.

To produce a well-defined instantiation of the biological model, we must generate a quantitative (mathematical) model which provides mathematical meaning to the biological concepts.

The next level converts the quantitative model into a computational model. In this conversion approximations may be introduced (for example Michaelis-Menten enzyme kinetics) and the model’s granularity (in space and time) is set. Decisions are also made as to how continuous functions in the mathematical model will be converted into computable functions, how boundaries and systemic quantities will be handled, and the various computational methodologies and solvers that will be employed.

The final step is the conversion of the computational model into simulation code suitable to be run in a particular simulation environment. The simulation code might consists of high level calls to modality specific APIs or might be lower level code written in, for example, C++.

In practice, this workflow often includes looping back from later stages to earlier stages. As the quantitative and methodological conversions are made there is often need to refine earlier stages in the model description. Often the quantitative model requires concepts and parameters not described in the original biological model. These discrepancies and omissions must be dealt with before proceeding to the next step.

And of course, there is another key feedback pathway; the output of the resulting simulation will, hopefully, be fed back to the biological model stage and extend our understanding of the biology.

Many simulation environments (and human model developers) jump directly from the Biological Model to the Simulation (Computational Model), but doing so mixes specific simulation methodologies with the biological and mathematical concepts of the model, destroying portability and obscuring important details and design choices. It also makes the utility of the biological model depend on the particular methodologies employed, so that development of novel simulation methodologies or computer architectures obsoletes the entire model. To maintain portability and to future proof the biological content of models against changes in hardware and software, we must separate the process of mathematical disambiguation of initial biological models from the translation of the quantitative description into a script for a specific computational modality.

To ensure transparency, mineability and shareability the Biological Model to Simulation Code process should follow a well-defined path and be described (and recorded) in an unambiguous way. To do this requires suitable language and process definition standards.

Abstraction Levels

To support the workflow in Figure 6 requires that model development proceed through well-defined levels of abstraction, with each level corresponding to specific languages and tools. The abstractions and languages are designed so that movement from more to less abstract model specification also automatically moves language organization from concepts represented using standard biological concepts to concepts organized around standard computational concepts.

General Structure of Language Standards, Tools and Toolkits

To be useful, a language standard requires a Toolkit with at least three components, the language standard itself, a set of tools for writing, editing, viewing and translating the language, and a set of use-cases to illustrate the language application (Figure 7). If the language is to be logically consistent, and be searchable for model-as-data applications, it also requires a reference ontology. The reference ontology must cover the important biological concepts needed in the use-cases.

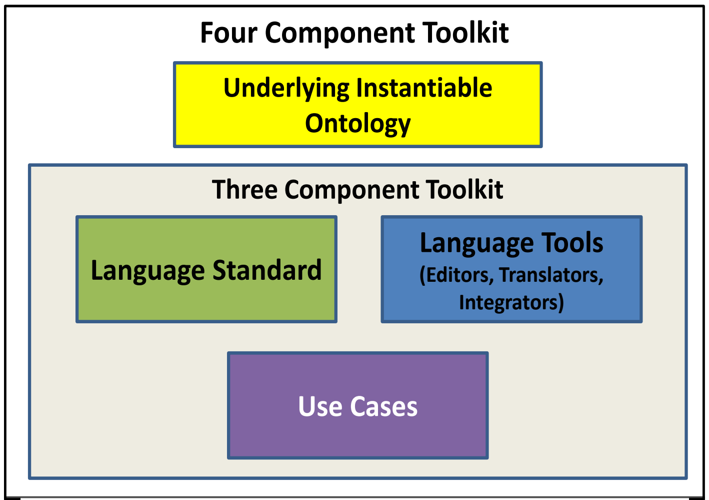


Figure 7: The elements of a three-component Toolkit and four-component language standard Framework.

The Process of Modeling: Levels of Model Description and Languages

Models and simulations consist of at least four elements; **objects**, **object properties / interactions**, **initial conditions** and **dynamics / processes** (Figure 8 & 8b). The execution of a simulation produces **data**, which may or may not have the same form as elements of the model or simulation. The results of a simulation might be a spatial structure which could also serve as an initial condition for another simulation, a kinetic constant which might be a parameter in an object property, or a new object, all of which might be used as input to a new model or to a model at higher or lower spatial scale.

Ultimately, we require languages to describe all of these elements, as well as meta-languages to describe their interactions. However, the number and nature of these languages are not immediately obvious. We can determine these by analyzing the workflow of developing a simulation and the simulation output.

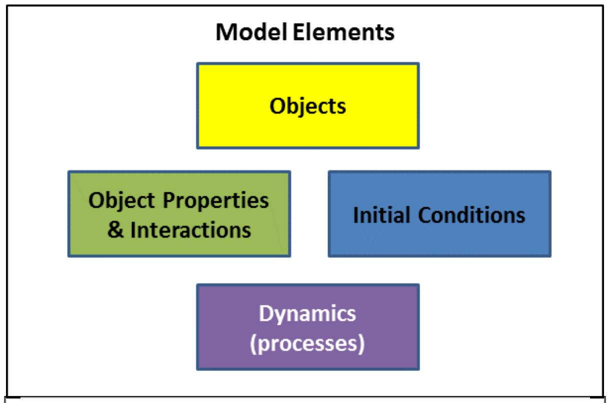


Figure 8: Basic Model Elements

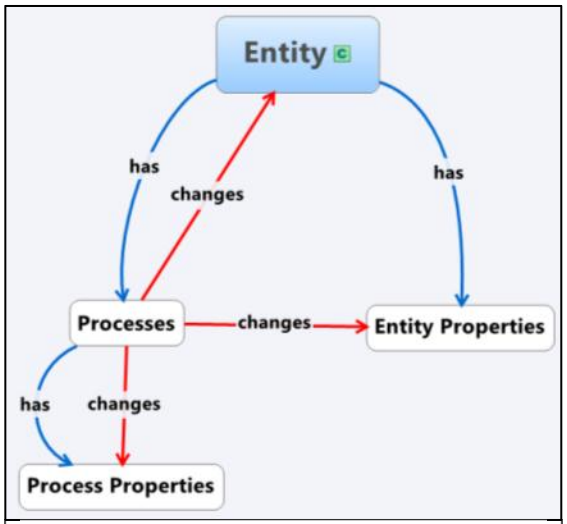


Figure 8b: Ontological Relationships for Model Description.

Simulation Output

If the output of the simulation is to be sharable and interpretable, it must be represented using one or more interchange languages (which might not be pure markup languages, which are often verbose for large data sets). The use of interchange languages to represent data allows simulation results to be compared to experiments, other simulations and to feed back to the higher levels of the modeling allowing model validation and refinement.

It is likely that as some level of detail it will become extremely difficult to completely describe the simulation output since the output may contain levels of complexity not present (or not recognized) in the original Biological Model.

Ontologies for Biology

A plethora of biological ontologies have already been developed. These ontologies range from exhaustive "naming authority" type ontologies such as FMA and GO to "toolkit" ontologies like SBML and OPB. The major difference between a "naming authority" and "toolkit" ontology is their ability to be used to *instantiate* a description of a particular biological system. Ontologies like FMA and GO provide a controlled vocabulary of concepts and generally place those concepts in some type of hierarchical tree. For example, FMA can be used to provide the name "hepatocyte" to the major cell type of the liver. In addition, the FMA trees locate that cell ("is part of" the "portal lobule" which "is part of" the "liver") and may also provide cell lineage or other information. However, FMA does not provide the terms needed to specify that a model might consist of multiple hepatocytes and those cells have a characteristic geometric distribution. In "naming authority" ontologies a new concept, for example a cell type in FMA or gene in GO, is generally added to the ontology and becomes a permanent component of that ontology.

"Toolkit" type ontologies are designed to provide a set of basic object types and a set of possible relations between those types. For example, to instantiate a hepatocyte in OPB one might describe the size and shape of the cell, its neighbors and the types of processes it can undergo. FMA does not already have a concept of “hepatocyte”, indeed it really doesn’t contain the concept of a cell. Once a cell is described using the terms of OPB the resulting description does not become a permanent part of the ontology.

We believe that MSM (and computational biology modeling in general) requires the creation of a suitable “mash-up” ontology that can be used to describe Biological Models and Computation Models.

Ontology development is a time consuming task. To develop the complete ontology needed for describing both wet-lab experiments and results and computational models is a daunting task. It may be possible to largely avoid the creation of a large and robust ontology by instead creating, essentially, a markup language and a fairly small ontology. The majority of the ontological terms (and relations) can be “inherited” from existing ontologies. For example, a model (wet-lab or computational) should use the excepted names for tissues, cells, genes and proteins. In a markup-up type description those terms can be extracted (linked to) existing “naming authority” ontologies such as FMA and GO. This implementation has several important advantages. 1) It automatically creates “crawlable” linkages to rich data repositories (not only the referent ontology but to linkages contained therein). 2) It insures that named entities in models use accepted names, which is a requirement for efficient mining of the models. 3) It significantly reduces the work required to develop a useable “model description” ontology. 4) It insures that the basic structure and concepts are “biologically” defined (as opposed to computationally defined).

For concepts relating to spatial, temporal and energy (K, k, ΔG …) terms OPB might be a suitable external ontology that can provide the needed terms and relationships.

The required depth of the “ontology” or mark-up language is unclear to us. It must be able to at least specify observable biology. This would include the ability to describe both objects (small molecules, cells, proteins, organs etc.) but also biological processes like transport, mobility, binding, development, differentiation, growth, death, signal generation and receiving and so forth. It seems likely that the language will also need to be able to describe the fundamental mathematics of a particular process. For example, kinetic and diffusion processes defined mathematically (but in a computationally agnostic way).

As the need to describe the model progresses from the Biological Model to the computational instantiation of the model the types of relationships may explode in number. It seems reasonable though that certain characteristic of the computational instantiation should be describable. For example, how spatiality is treated. Is space discretized or continuous? Does the model even have a concept of space? The same for time, are there time steps (if so how big) or not? How exactly are fundamental biological processes instantiated? If the model includes cells that grow what controls the growth and what formula is used to calculate the growth as a function of time?

Even if it is found that an ontological description of all of the computational details is not practical it is still critically important that the Biological Model is described using the structured language(s) down to at least the API level

Some work has already been done on “mash-up” type ontologies for computational biology. SemSim and SemGen (Univ. Washington) include many of the ideas we have presented here. Extending those tools to describe wet-lab assays and results and enhancing their ability to describe Biological Models would leverage existing software and may provide a rapid route to a functioning ontology and markup language tool. Adding the ability to publish the models via the SW may provide a rapid route to model dissemination, validation, mining and reuse.

Advantages of an “Ontological” description at the Biological Model and other levels

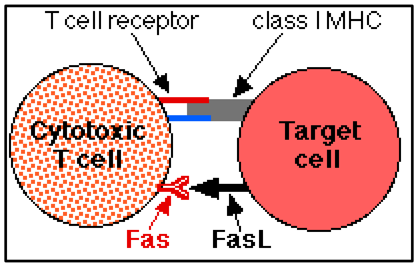
If models (biological, computational …) can be described in a structured way then the models can be published using semantic web (SW) technologies[2]. This method of publication makes the models searchable without a user needing to know where to actually look for the models. The model might be in a well maintained computational model repository (like the BioModels Database), or it might be on a server maintained by an individual research lab. The data might be a Biological Model, or a computational model, or an entry in a vast repository of biological assay results such as the EPA’s ToxCast system. (Which should be describable using our imagined ontology and publishable via the SW, instead of being buried in a web accessible database which cannot be automatically mined.)

An important benefit of our imagined ontology is that experimental data is describable, SW publishable and searchable (mineable). Since that data is often used to parameterize models finding it should become easier.

Defining the Ontology (or markup language)

The ontology perhaps should be developed in two arms. The first arm would be the selection (possibly with certain restrictions) of the reference ontologies. The second arm is the identification of key terms and relationship that are either absent in the reference ontologies or that are of such fundamental import to experimental and computational biology that a redefinition is warranted.

The key terms and relationships could perhaps be defined using two pathways. One pathway would start with wet-biology and biologist. What are the fundamental objects and processes that they think must be included? The second pathway would start with existing computational biology models and work backwards. It is not necessary that the ontology can describe all the details of the computational model but it appears to us that most computational biology groups have already (if unconsciously) followed the first pathway and there modules and code have a tendency to mirror basic biological concepts. Working backwards from exiting computational models leverages that knowledge.



A Really Simple Sample Model and Markup

What exactly would the process look like and what would it create? We will start with the very simple model, like what might be found in a basic biology text book. Shown at the right is the interaction of

[2] The semantic web has not been widely accepted as a general method of coding information on web pages. However, in this case only the pages that our community generates need to conform to the standard.

a Cytotoxic (killer) T-cell with an infected host cell. (From http://users.rcn.com/jkimball.ma.ultranet/-BiologyPages/T/Transplants.html)

We start with describing the objects in the model, two cell types and four membrane bound proteins. We can use FMA to describe (unambiguously name) the cells and GO to describe (unambiguously name) the four proteins. We can use terms and relationship from OPB to describe the binding events. In pseudo XML like format[3];

```
<XML="MSM pseudo Code">
<External "MSM" link to MSM>  <!—our imaginary ontology -->
<External "FMA" link to FMA>
<External "CL"  link to CL >
<External "GO"  link to GO >
<External "OPB" link to OPB>
<Cell type=FMA:"PREFERRED NAME=T-cytotoxic cell" FMA:"FMAID=70573">
       <protein location=FMA:" Plasma membrane" name=GO:"T cell receptor">
       <protein location=FMA:" Plasma membrane" name=GO:"Fas">
</Cell>
<Cell type= FMA:"PREFERRED NAME=cell" FMA:"FMAID=68646">
       <protein location=FMA:"Plasma membrane" name=GO:"Class I MHC">
       <protein location=FMA:"Plasma membrane" name=GO:"FasL">
</Cell>
<Interaction type=binding>
       <entity1 name=GO:"T cell receptor">
       <entity2 name=GO:"Class I MHC">
       <FMA:Dissociation Constant="1e-7" Units="Molar">
</Interaction>
<Interaction type=binding>
       <entity1 name=GO:"Fas">
       <entity2 name=GO:"FasL">
       <FMA:Dissociation Constant="1e-6" Units="Molar">
</Interaction>
```

The markup above adequately recapitulates the details of this very simple Biological Model. Though simple the model still contains a significant amount of information ("knowledge"). To extend this markup to a particular computational framework might involve including terms to describe how the binding is calculated; mass action law, stochastic etc. A particular computational framework might also need to describe the movement of the cells and that requires some definition of space, distance and time. However, regardless of the markup that is added to describe the computational instantiation of the model the Biological Model, and its description, is constant.

Besides describing the basic model the markup above provides "crawlable" linkages to other information. For example, following the FMA linkage for FMA:"PREFERRED NAME=T-cytotoxic cell" provides the synonyms for this cell type: "Cytotoxic T-lymphocyte", "Cytotoxic T cell", "Killer T lymphocyte", "Killer T cell" and "Cytotoxic T lymphocyte". Following the GO linkage for "Class I MHC" leads to gene and protein data such as MW, sequence and tissue distribution.

The sample markup above gives a very simple pseudo-description of the binding interaction. The ontology should also allow more detailed models to be incorporated into the model's description. As long as there is a systematic way to describe the components, preferably via linkages to existing ontologies, then any markup can be included in the model. For example, an SBML type model could be included to describe the computational modality for the binding equilibria.

[3] The style should probably by OWL/RDF instead of simple nested XML.

A More Complex Multiscale and Multimodal Model

Figure 9 presents a simplified view of a MSM of embryonic somitogenesis (Belmonte 2010). In this model the precursors to the vertebrae are being formed in an embryo from a proliferating pool of pre-somitic cells, which differentiate and self-organize under the control of a molecular clock. At the lowest scale a set of ODEs describe the transcriptional behavior and activity of the genes involved in the clock. The gene expression behavior modifies the behavior at the cellular level. Secretion of diffusible signal molecules, modeled using PDEs, provide communication with both proximal and distal cells. The cellular differentiation decision is made based on a Boolean tree.

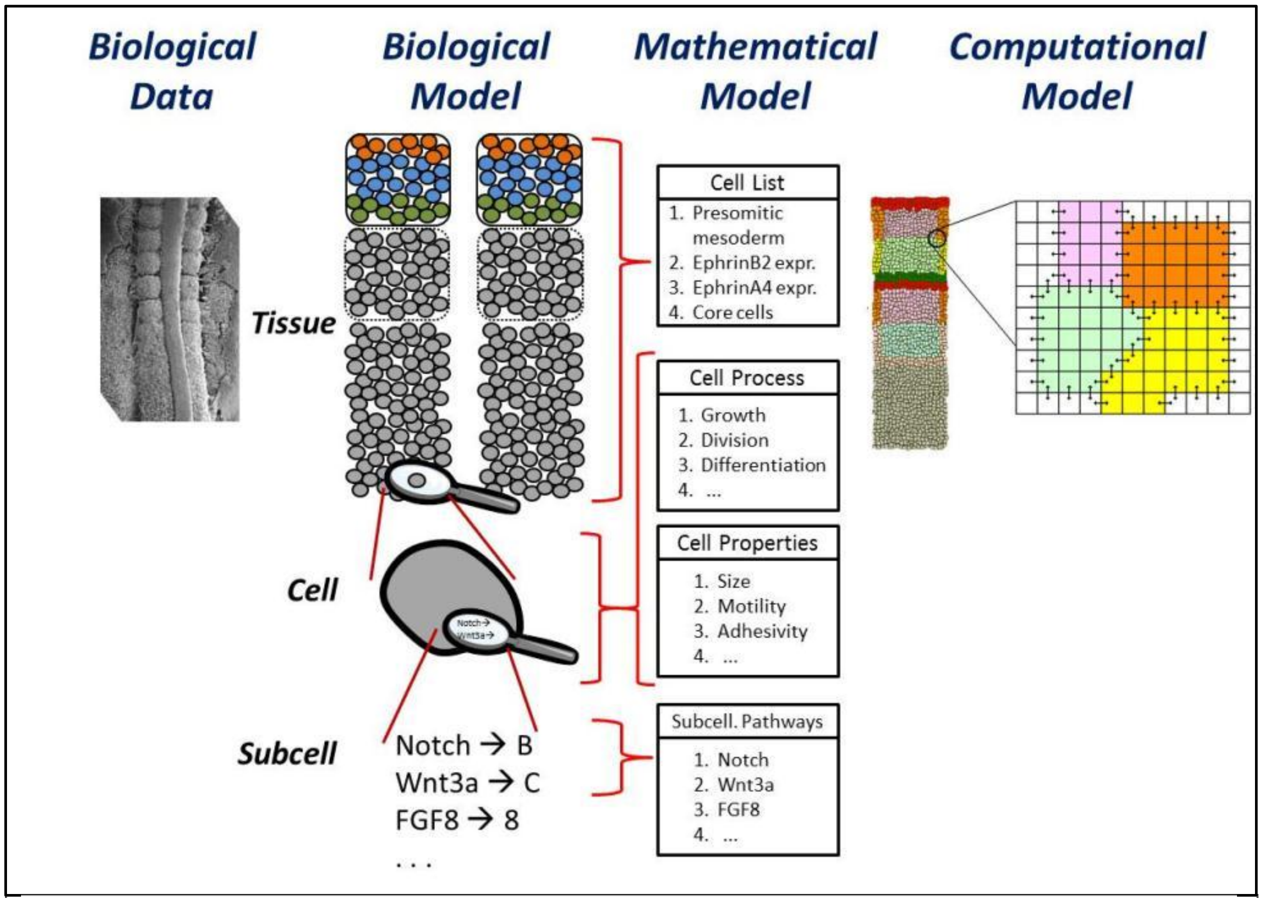


Figure 9: Multiscale model definition workflow for embryonic somitogenesis showing the formation of cell clumps which are precursors to vertebrae.

From left to right in Figure 9 is the progression from biological data, to Biological Model, to Mathematical Model to Simulation Code. As one proceeds from left to right the description of the model becomes less biological and more computational. As one proceeds from the top to the bottom the scale of the model decreases. At the top, the model spans a centimeter or so of the developing embryo, at the bottom the model describes subcellular gene and gene product scale events.

Model Sharing and Reuse

An effective path to preparing a multiscale model would include identifying existing models that contain useable components for the multiscale model. Identification of suitable existing model would be greatly facilitated if those models include biological descriptions created with suitable reference ontologies. In particular, the use of "naming authorities", such as FMA and GO, insures that the named entities adhere to accepted naming conventions. Misnamed entities in model descriptions effectively become invisible to most search technologies.[4] In addition to providing consistent names, the reference ontologies can provide higher levels of abstraction that will greatly facilitate identification of models that are conceptually identical even though the names of the actual entities are different.

For example, consider the simple two cell model of a Killer T-Cell interaction outlined earlier. If that description was in a searchable database (or SW form) then it could be located with queries such as "T-cytotoxic cell" or "T cell receptor". In addition, since "T-cytotoxic cell" is not the commonly used name, a semantic web search engine with access to FMA would also be able to locate this model if the user used any of the pseudonyms that FMA provides such as "Cytotoxic T-lymphocyte", "Cytotoxic T cell", "Killer T cell" etc.

Besides provide a consistent naming and name resolution mechanism the use of structured reference ontologies also makes it possible to search for models using more general terms. In the above example "T-cytotoxic cell" **ISA** "cell" and "T cell receptor" **ISA** "cell membrane bound receptor". That means that this ontological description, and any associated computational model, would be identified as a generic model of a cell with a membrane bound receptor. In addition, this type of abstraction will easily identify equivalent models implemented in different computational modalities. This high level of abstraction would be expected to greatly facilitate the identification and reuse of model components.

Existing Approaches to Cross scale (and cross modality) Model Descriptions

SemSim (Gennari 2008) is a tool and methodology that facilitates the back mapping of a Simulation Code to appropriate reference ontologies. Once the model is back mapped the terms can then be forward mapped into other code modules. SemGen (Gennari 2010) is a graphical interface that facilitates generating composite annotations using multiple ontologies. SemSim, SemGen and related tools, provide proof of concept studies of the power of using well defined ontologies and mapping tools to combine disparate models.

Long Term Need for a Robust MSM Infrastructure

As single and multiscale biological modeling advances from a basic research tool to a tool that can be used for medical and/or regulatory decision making, the requirements for a robust infrastructure will increase significantly. For example, to use an MSM model to guide the treatment of a patient in a clinical setting will require a rigorous, and perhaps even onerous, means of tracking the computational code, parameters and patient specific characteristics of the model. In the area of government regulation of environmental toxins, MSM models may be used to guide the regulatory decision making process (NRC 2007). In both the medical and regulatory cases the model, code(s), parameters etc. must be retained

[4] In the Biomodel database conversion of uncurated models to curated models often includes corrections in the names of entities like genes and proteins. This process should have been done by the original model builders. Not doing so reduced the validity and usability of their computational model.

long term and the computational process must be "transparent". The regulatory decision making process must be defendable and reproducible even years after the decision was made (Pascual 2009). In both of these long term applications of MSM, irreproducibility of the computational model, loss of needed code and parameters, and the loss of the decision making process that went into the design of the computation will be unacceptable. The use of MSM as a pure research tool will greatly benefit from suitable versioning and archiving of all aspects of the development and use of computational models. In the medical and regulatory domains these aspects will become requirements instead of just an advantageous tool. It would be advantageous to start to develop these standards now.

Needs

Figure 10 adapts the Physiome framework to include possible reference ontologies. In addition to existing ontologies, such as FMA and GO, it is likely that additional ontologies will be needed, particular for the various Computational Modalities.

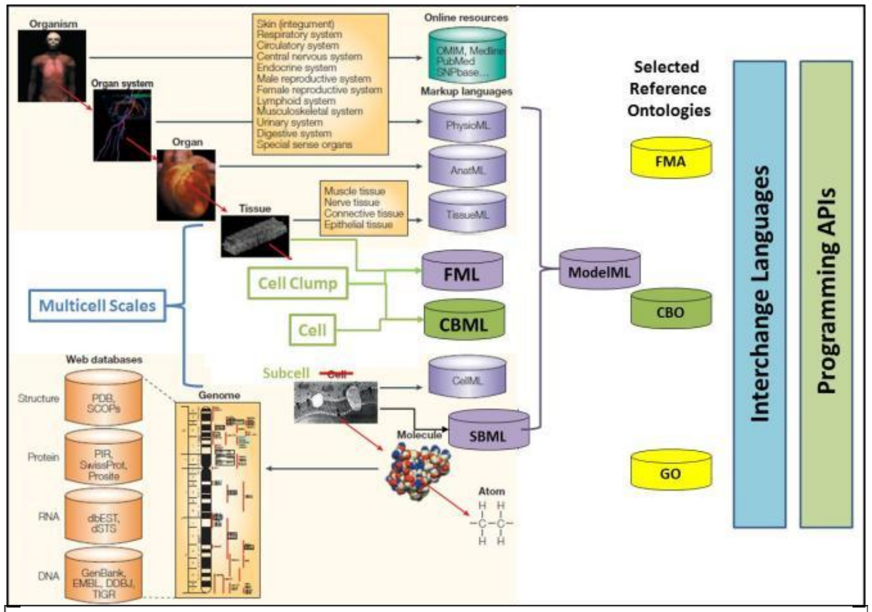


Figure 10: Biological scales and the corresponding computational tools augmented with additional reference ontologies, exchange languages, and APIs to form a Model Development Framework.

In addition, tools that map from the ontological description to particular computational tools will be needed. In Figure 10 we have added the interchange languages and programming APIs and have indicated that they must span the entire range of scales.

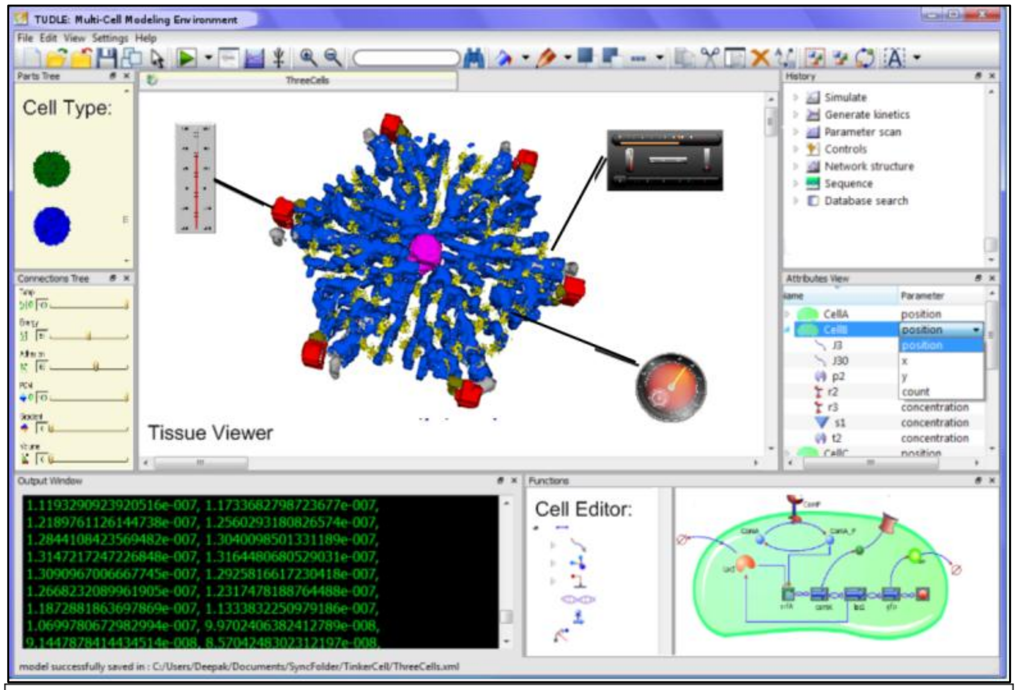
Finally, to actually get users to *use* this method of model definition a tool is needed that provides **immediate tangible benefit** to the user. A suitable "carrot" might be a graphical user interface that allows biologists, both wet-lab and computational, to describe their models. The user would immediately benefit from having a high quality graphical description of their model and would also have a searchable (mineable, sharable…) description of their model. The tool could greatly simplify, and hence "enforce", the use of the appropriate reference ontologies (or markup languages) at all stages of the model building process. A tool that provides both easy markup as well as other immediate tangible results will greatly increase the chances that the markup and ontology tools are widely adopted.

Overall, we are talking about developing the infrastructure required to support and share biological models that span multiple scales and incorporate diverse computational modalities. Specific needs include:

1. Use cases for;
    a. Various modeling modalities (FEA, PBPK, ODE, …)
    b. Various scales (organism, tissue, cell, sub-cell)
2. Selection of suitable existing ontologies. (Note that existing ontologies often have significant overlap, which must be resolved in some manner.)
    a. Foundational Model of Anatomy (FMA), technically limited to humans.
    b. Cell Ontology.
    c. Gene Ontology (GO).
    d. Medical Subject Headings (MeSH), strong in abnormal, disease and medical biological terms.
    e. Ontology of Physics for Biology (OPB).
    f. Chemical Entities of Biological Interest (ChEBI) for small molecules of biological interest.
    g. Description of small molecule to macromolecule interactions. (For example Choi 2010.)
    h. Description of metabolic conversion of small molecules. (For example Sankar 2007)
3. Creation of any missing ontologies specific for biological and multiscale modeling (biomechanical, FEA, PBPK, Cell based, …).
4. Search use cases for mining of the SW/ontological model storage format.
5. Search engine suitable for the SW repositories.
6. Proof of concept modeling platforms. (Use cases for integration of particular computational modalities.)
7. A well developed, open source, extensible tool ("carrot") that facilitates model description, SW publication and provides an immediate tangible benefit to the user, while at the same time ensuring good model definition practices.

The final need in the above list is perhaps the most important. We believe there is a critical need to develop a user friendly tool that facilitates model definition (in a consistent format). The tool would be the "martialling point" for the integration of solid ontological descriptions of the biology being modeled, the definition of the modeling modality and the description of the model output.

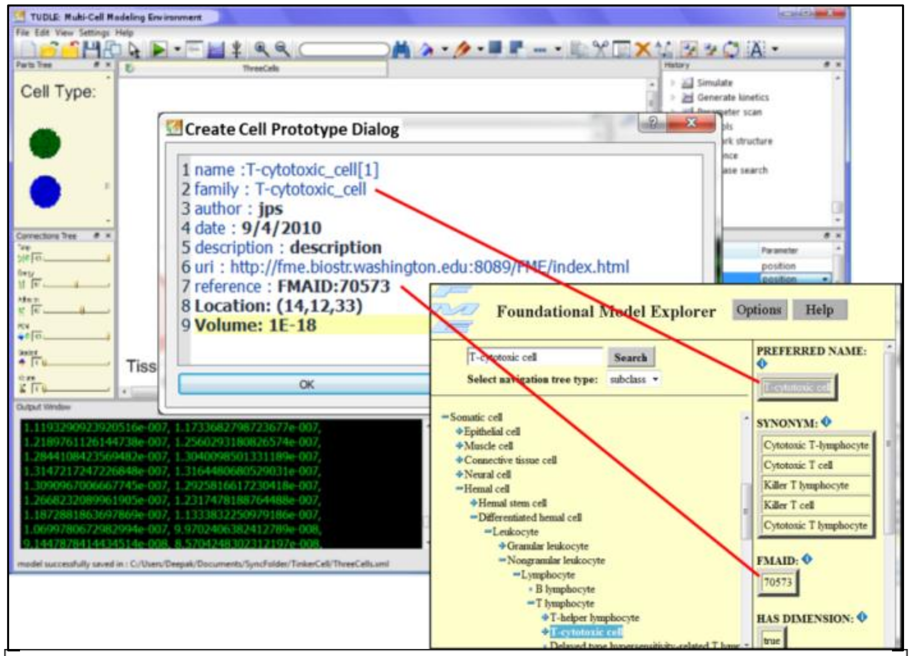
We envision a tool similar, but significantly enhanced, to TinkerCell (Deepak 2009). The tool should provide a graphical interface to lay out components of the model, in both two and three dimensions. Predefined components would range from whole animals, to tissues, to cell clumps, to cells, to sub-cellular, to molecular domains. Browsers windows to the various reference ontologies would intelligently guide the user to the appropriate ontology where the proper name or concept can be selected. For supported computational modalities, dialogs would guide the user through the conversion of the biological model to the mathematical model to the computational model and ultimately to the executable code. A mock-up of what the interface might look like is shown in figures 11a and b.



```
1.1193290923920516e-007, 1.1733682798723677e-007,
1.2189761126144738e-007, 1.2560293180826574e-007,
1.2844108423569482e-007, 1.3040098501331189e-007,
1.3147217247226848e-007, 1.3164480680529031e-007,
1.3090967006667745e-007, 1.2925816617230418e-007,
1.2668232089961905e-007, 1.2317478188764488e-007,
1.1872881863697869e-007, 1.1333832250979186e-007,
1.0699780672982994e-007, 9.9702406382412789e-008,
9.1447878414434514e-008, 8.5704248302312197e-008,
```

model successfully saved in : C:/Users/Deepak/Documents/SyncFolder/TinkerCell/ThreeCells.xml

Figure 11a: Mock-up of the "carrot". The main window shows a liver lobule modeled at the cellular level. The lower right window shows the sub-cellular SBML model for an individual cell.


TUDLE: Multi-Cell Modeling Environment
Create Cell Prototype Dialog
OK
Foundational Model Explorer
Options
Help
Search
Select navigation tree type:
subclass
PREFERRED NAME:
T-cytotoxic cell
SYNONYM:
FMAID:
HAS DIMENSION:

1 name :T-cytotoxic_cell[1]
2 family : T-cytotoxic_cell
3 author : **jps**
4 date : **9/4/2010**
5 description : **description**
6 uri : http://fme.biostr.washington.edu:8089/FME/index.html
7 reference : **FMAID:70573**
8 **Location: (14,12,33)**
9 **Volume: 1E-18**

T-cytotoxic cell

- Somatic cell
  - Epithelial cell
  - Muscle cell
  - Connective tissue cell
  - Neural cell
  - Hemal cell
    - Hemal stem cell
    - Differentiated hemal cell
      - Leukocyte
        - Granular leukocyte
        - Nongranular leukocyte
          - Lymphocyte
            - B lymphocyte
            - T lymphocyte
              - T-helper lymphocyte
              - T-cytotoxic cell
              - Delayed type hypersensitivity-related T lymp

SYNONYM:
Cytotoxic T-lymphocyte
Cytotoxic T cell
Killer T lymphocyte
Killer T cell
Cytotoxic T lymphocyte

FMAID: 70573

HAS DIMENSION: true

Figure 11b: Mockup of the “create cell” dialog showing linkage to the appropriate ontology for proper naming of the cell.

Conclusions

MSM is at a turning point in its development. As models become more complex, more robust, and more useful it is becoming clear that there will not be a "one size fits all" approach. Multiple models, instantiated in multiple computational modalities and spanning multiple scales will most likely be the long term paradigm in MSM. This heterogeneous approach presents unique problems in model definition, validation, storage, mining and reuse. As MSM moves from the basic research domain into the regulatory and medical domains it will become increasingly important that models are documented, transparent and reproducible. We believe these requirements require a robust, defensible, logical and publishable method of describing all phases of biological model development. In the short term, a suitable descriptive language, and the tools needed to use that language, will significantly help the MSM community develop "best practice" standards. In the long term those standards will be the basis for the much more exacting requirements that the longer term medical and regulatory applications will require.

References

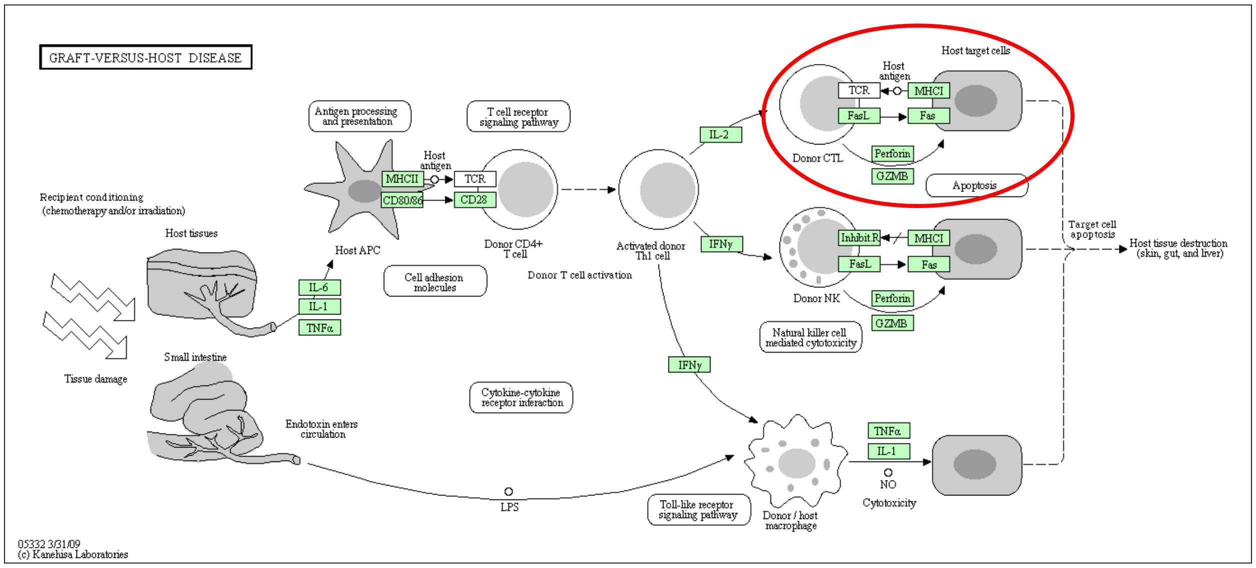
1. **Ashburner M, *et al.*, (2000),** "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium", Nat Genet. 2000 May;25(1):25-9. GO
2. **Bard** J, Rhee SY, Ashburner M., (**2005**) "An ontology for cell types", Genome Biol. 2005;6(2):R21. Cell Ontology
3. **Belmonte** JM, Hester, SD, Clendenon S, Gens JS, Glazier JA (**2010**), "Modelling of the Interaction of Oscillatory Netwroks and Biomechanics in Somitogenesis", *in preparation*.
4. **Chang**, David Chan-Wei**,** PhD Thesis, **2010**, "A Computational Framework to Describe and Solve Temporo-Spatial Biological Models", University of New South Wales. (James reviewed)
5. **Choi**, JY, Davis, M. J., Newman, A. F., Ragan, M. A. (**2010**) "A Semantic Web Ontology for Small Molecules and Their Biological Targets", J. Chem. Inf. Model, 50, 732-741.
6. **Cook** DL, Mejino JL, Neal ML, Gennari JH., (**2008**) "Bridging biological ontologies and biosimulation: the ontology of physics for biology", AMIA Annu Symp Proc. 2008 Nov 6:136-40. OPB
7. **Deepak** C, Bergmann FT, Sauro HM, (**2009**) "TinkerCell: modular CAD tool for synthetic biology", Journal of Biological Engineering, 3:19.
8. **Gennari**, J. H., Neal, M. L., Galdzicki, M., Cook, D. L., (**2010**), "Multiple ontologies in ation: Composite annotations for biosimulation models", J. Biomed Inform, doi:10.1016/j.jbi.2010.06.007. SemGen
9. **Gennari**, J. H., Neal, M. L., Carlson, B. E., Cook, D. L., (**2008**), "Integration Of Multi-Scale Biosimulation Models Via Light-Weight Semantics", Pacific Symposium on Biocomputing 13:414-425. SemSim

10. **Hucka**, M. et al., (**2003**) “The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models”, Bioinformatics, 9(4):524–531. SBML
11. **NRC** (National Research Council) (**2007**). Toxicity Testing in the 21st Century: A Vision and a Strategy. Washington, DC, National Academies Press.
12. **Pascual** P, (**2009**) “Evidence-based decisions for the wiki world”, International Journal of Metadata, Semantics and Ontologies, 4:4, 287-294.
13. **Rosse** C, Mejino JL Jr., (**2003**) “A reference ontology for biomedical informatics: the Foundational Model of Anatomy”, J Biomed Inform. Dec;36(6):478-500 FMA
14. **Roux-Rouquie** et al., **2005**, “Metamodel and Modeling Language: Towards an Unified Modeling Language (UML) Profile for Systems Biology”, SCI05, Orlando, Florida, USA, July 10-13.
15. **Sankar**, P., Ahila, G. (**2007**) “Ontology Aided Modeling of Organic Reaction Mechanisms with Flexible and Fragment Based XML Markup Procedures”, J. Chem. Inf. Model, 47, 1747-1762.
16. **Sauro**, H.M., Hucka, M., Finney, A., Wellock, C., Bolouri, H., Doye, J., Kitano, H., (**2003**) “Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration. OMICS.” Winter; 7(4):355-72.
17. **Shirinifard** A, Gens JS, Zaitlen BL, Popławski NJ, Swat M, et al. (**2009**) “3D Multi-Cell Simulation of Tumor Growth and Angiogenesis”, PLoS ONE 4(10): e7190.
18. **Sun**, Zhouyang, PhD Thesis, **2008**, “Using Ontology and Semantic Web Services to Support Modeling in Systems Biology”, Centre for Mathematics & Physics in the Life Sciences and Experimental Biology, University College London.

Appendix I: Models Starting From a Biologists Perspective Continued

A More Complex Sample Model and Markup

The KEGG path way (http://www.genome.jp/kegg-bin/show_pathway?map05332) below is an example of a more complex Biological Model.


GRAFT-VERSUS-HOST DISEASE
Recipient conditioning (chemotherapy and/or irradiation)
Host tissues
Tissue damage
Small intestine
Endotoxin enters circulation
LPS
IL-6
IL-1
TNFα
Host APC
Antigen processing and presentation
T cell receptor signaling pathway
Host antigen
MHCII
TCR
CD80/86
CD28
Donor CD4+ T cell
Cell adhesion molecules
Donor T cell activation
Activated donor Th1 cell
IL-2
IFNγ
Cytokine-cytokine receptor interaction
Toll-like receptor signaling pathway
Donor / host macrophage
Host target cells
Host antigen
TCR
MHCI
FasL
Fas
Perforin
GZMB
Donor CTL
Apoptosis
Inhibit R
Donor NK
Natural killer cell mediated cytotoxicity
Target cell apoptosis
Host tissue destruction (skin, gut, and liver)
TNFα
IL-1
NO
Cytotoxicity
05332 3/31/09
(c) Kanehisa Laboratories

In this model there are several cell types and about a dozen proteins. This model contains as a sub-model the very simple model we examined earlier; the recognition of an infected cell, via the T-Cell Receptor to Class I MHC interaction, by a cytotoxic T-Cell (circled in red). Note that the names used in this KEGG figure do not exactly match the names used in the first model. To make a model sharable it is critically important that this type of discrepancy is avoided. The use of reference ontologies when the model is described in our markup/ontological description regularizes the names and makes it possible to identify the first model as a sub-model of the second. In other words, consistent naming within the model description makes it possible to find candidate existing models that can be reused.

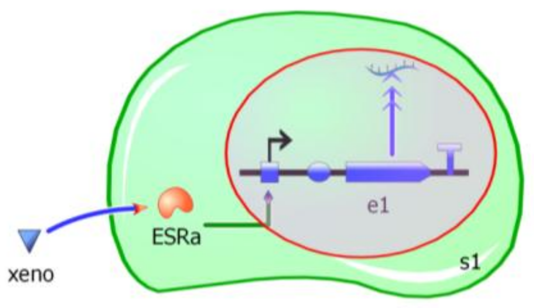
The variability in the naming of cells, tissues, proteins and genes is a significant barrier to identifying and reusing models.

A Completely Different Type of Sample Model

It should be possible to encode simple Biological Models that are run as screens, including cases where the screens are high-throughput. For example, an Attagene Inc. assay being run for the EPA measures the transcriptional activation effects of xenobiotics binding to the Estrogen Receptor Alpha. A very small fragment of the data table for the Estrogen Receptor Alpha assay is shown below.

# Data Set Name: Attagene					
# Column 1: Unique chemical identifier (320 unique values)					
# Column 2: CAS Registry Number (309 unique values)					
# Column 3: Chemical name (309 unique values)					
# Columns 4-N: Assay data. For a description of the assays, see the Assay Definitio					
# Values are LEL (lowest effective level)					
# Values are in microM					
# Inactive chemical-assay combinations are indicated by a value of "-"					
SOURCE_NAME_SID	**CASRN**	**NAME**	**Ahr_CIS**	**AP_1_CIS**	**BRE_CIS**
DSSTOX_40310	136-45-8	2,5-Pyridinedicarbox	-	-	-
DSSTOX_40542	90-43-7	2-Phenylphenol	-	-	58
DSSTOX_40375	55406-53-6	3-Iodo-2-propynylbut	-	-	-
DSSTOX_40294	135158-54-2	Acibenzolar-S-Methyl	38	-	-
DSSTOX_40338	50594-66-6	Acifluorfen	-	-	-
DSSTOX_40339	15972-60-8	Alachlor	-	7.4	12
DSSTOX_40344	33089-61-1	Amitraz	-	-	-
DSSTOX_40299	101-05-3	Anilazine	62	45	-
DSSTOX_40347	86-50-0	Azinphos-methyl	-	44	23
DSSTOX_40348	131860-33-8	Azoxystrobin	-	3.8	37

This is a cell based assay and the compounds tested must cross, at least, the cell membrane. The Biological Model might be represented as;

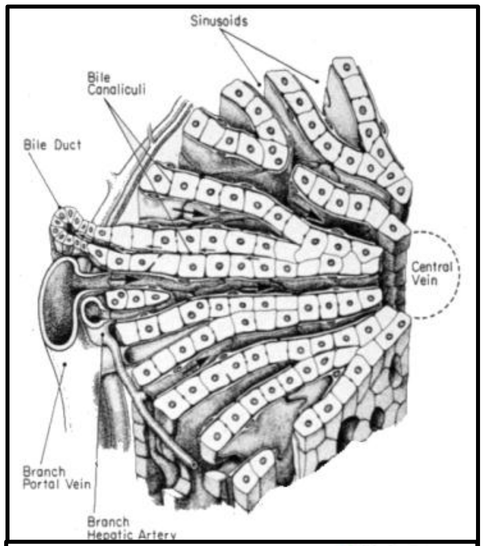


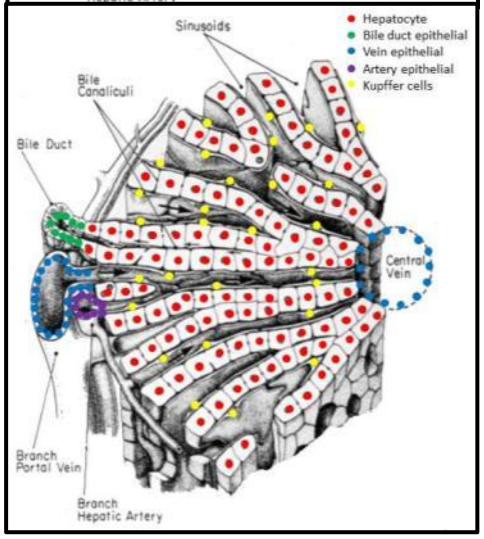
"Xeno" represents a single compound from the assay table. The molecular characteristics of "Xeno" can be obtained from other tables associated with the EPA ToxCast system. (ToxCast is not SW compatible but hopefully will be eventually.)

```
<XML="MSM pseudo Code">
<External "MSM" link to MSM>  <!—our imaginary ontology -->
<External TCC link to ToxCastCompounds >
<External TCA link to ToxCastAssays >
<External "GO"  link to GO >
<Cell type=FMA:"PREFERRED NAME=cell" FMA:"FMAID=68646">
        <protein location=FMA:"cytosol" name=GO:"ESRA">
</Cell>
<Interaction type=binding>
        <entity1 name=GO:"ESRA">
        <entity2 name=TCC:"Name=Acibenzolar-S-Methyl" id=TCC:"CASRN=135158-54-2">
        <createdEntity>entity1:name+entity2:name</createdEntity>
        <MSM:LEL="38" Units="microMolar">
</Interaction>
<transport>
        <entity name=TCC:"Name=Acibenzolar-S-Methyl" id=TCC:"CASRN=135158-54-2">
        <type = passive>
        <cross= type=FMA:"PREFERRED NAME=cell" FMA:"FMAID=68646" FMA:"cell membrane">
</transport>
<transport>
        <entity name=entity1:name+entity2:name>
        <type = active>
   … describe translocation to the nucleous of the complx
</transport>
<Interaction type=binding>
        <entity1 name=MSM: entity1:name+entity2:name >
        <entity2 name=GO:"Gal4" type=GO:CDS>
```

```
</Interaction>
```

Once the Biological Model is described it can easily be changed to represent other rows in the assay results table by simply changing the compound reference and the results value.





Another Completely Different Type of Sample Model

Using our imagined ontology and mark-up language it should be possible to describe tissues and organs. At right is a drawing of a liver lobule. In this drawing there are two main cell types shown; Hepatocytes and Kupffer cells (liver macrophages). In addition the central and peripheral blood vessels are shown diagrammatically. In the case of the liver, the arrangement of the cells forms the sinusoids through which blood flows from the periphery to the central vein of the lobule. Blood flow through the sinusoids mixes venous blood from the GI tract with oxygenated blood from the hepatic artery. A Biological Model of the liver might include both the cell types and their spatial distribution along with a description of the blood flow pattern.

As with the previous examples, development of the Biological Model should start with a description of the cell types involved (with correct linkages to a naming authority such as FMA or CL) along with spatial descriptors defining each cell. In addition, a description of the blood flow (with "blood" also properly identified using, for example, FMA) would also be of use. The required level of detail describing the blood flow might range from simply indicating that the periphery is at higher pressure than the central vein, or the description might include terms such as viscosity and the cross-sectional area of the sinusoids (described using terms from OPB).

This particular model also introduces the need to be able to describe adhesion between cells. In the case of the liver, cell-cell adhesion is responsible for maintaining the basic topology. Adhesion could be described in the Biological Model as simple "stickiness" between the pair of cells. Or, if there is sufficient knowledge about the interaction the adhesion might be specified using explicit adhesion molecules.