ibres volume 29, issue 2, pages 109-122 (2020)

Semantic Simulations Based on Object-Oriented Analysis and Modeling[1]

Robert B. Allen
0000-0002-4059-2587
rba@boballen.info

ABSTRACT

Background. This is part of an ongoing set of studies to develop a framework and techniques for rich semantic modeling, based on traditional ontologies and upper ontologies. Such models should provide the basis for direct representation of complex texts such as scientific research reports and descriptions of historical events. Semantic models have been developed to describe mechanisms and explore how semantic models can be related to object-oriented programming languages.

Objective. Object-oriented semantic modeling was applied to two simulations: of a waterfall and of the cardiopulmonary system. Issues related to the components of these models and their interaction were examined.

Results. Complete executable models for the waterfall and the cardiopulmonary system examples are demonstrated. Specific recommendations are offered for handling States, Portions of Matter, Meta-Operators, and Systems.

INTRODUCTION

We have proposed direct representation as a strong form of information organization. Direct representation proposes that rich semantics can support indexing and access by developing structured models to supplement or even replace traditional text. For instance, community models might be implemented which could help to orient readers of historical newspapers from a town or region. As a different sort of application, we envision highly structured versions of scientific research reports. Potentially, models could be developed for different aspects of research reports such as hypotheses about the phenomenon under investigation, research workflows, data collection and data sets, and structured argumentation about the implications. Further, all of these could be coordinated into a unified framework as a fully machine-accessible version of the research-reports.

As a first step in developing such models, we build on upper ontologies such as BFO, DOLCE, and YAMATO. These upper ontologies describe the types of entities allowed for domain ontologies that are based on them. We take upper ontologies as providing a form of data typing (Allen & Kim, 2018), and we apply the structures provided by those upper ontologies to implement a *model layer* which emphasizes the interaction of the entities. Moreover, our semantic modeling goes beyond traditional modeling to additional components such as complex objects with parts and states, mechanisms, and a microworld in which interactions take place. The models, then, are executable semantic simulations based on what is, essentially, an object-oriented programming language. In previous work, we have

[1] This paper is based on a presentation at the Workshop on Rich Semantics and Ontology, Kuala Lumpur, Nov. 2019, and which appeared as arXiv: 1912.13186.

Published by WKW School of Communication & Information & NTU Libraries, Nanyang Technological University

suggested the value of object-oriented analysis with rich semantics to the systematic description of natural systems. This is supported by Galton and Mizoguchi's (2009, p. 1) definition of objects: "… we identify an object as an interface between those processes which are internal and those processes which are external to it …" As the primary goal in this paper is to explore issues in implementing the rich semantic models, we:

- present Structured Definitions of a Waterfall
- provide an Executable Semantic Model of the Cardiopulmonary System
- describe Parts of Objects
- consider the Behavior and States of Parts of Objects
- discuss Functions, Mechanisms, Systems, and Microworlds
- describe Modeling
- consider Validation and Extrapolating Mechanisms as well as Semantic Modeling for scholarly communication.

STRUCTURED DEFINITIONS

Waterfall Example

The philosopher Heraclitus famously said: "You can never step into the same river twice." He concluded that "all is change." By comparison, other philosophers emphasize the static nature of objects. Still others adopt a hybrid that is also consistent with natural language, which has both verbs and nouns. When one is running, one is changing position in space. Nonetheless, Heraclitus's river is still widely discussed. The action of flowing is integral to the essence of what it means to be a river. A version of this quandary focusing on waterfalls has recently been discussed (Galton & Mizoguchi, 2009).

According to dictionary definitions, a waterfall is:

> *… a cascade of water falling from a height, formed when a river or stream flows over a precipice or steep incline*[2]

and,

> *an area where water flows over a vertical drop or a series of steep drops in the course of a stream or river.*[3]

Figure 1 shows a structured definition for a waterfall that covers the key features of the definitions. We have incorporated specific parameters, so it is executable and provides a simple simulation of a waterfall. There is a class for the Waterfall itself and two other classes for the main parts of a waterfall: the *StreamPath* and the *Water*. In the program, the water flows in the stream bed with a small slope and then comes to a steep drop. Because of the constructor (*__init__*), the water starts to flow as soon as the program is initialized. The water is modeled as individual *Portions* and is assigned to different location states as it flows (e.g., upper stream, drop, pool) (cf., Hornsby & Cole, 2007).

Commentary on the Waterfall Example

Defining objects with object-oriented classes highlights and provides a structure for resolving the issues raised by philosophers and lexicographers. The ongoing

[2] https://www.lexico.com/en/definition/waterfall
[3] https://en.wikipedia.org/wiki/Waterfall

```
class StreamPath():
def TraverseUpperStreamBed(self,i):
 for L in range(0,self.upperBedLength):
  self.water[i].X = self.water[i].X+10
  self.water[i].Y = self.water[i].Y-1
  self.water[i].Location="upper"
def TraverseDrop(self,i):
 for V in range(0,self.verticalDrop):
  self.water[i].X = self.water[i].X+1
  self.water[i].Y = self.water[i].Y-10
  self.water[i].Location="drop"
def Pool(self,i):
 self.water[i].Location="pool"
def __init__(self, water):
 self.water = water
 self.upperBedLength=1000
 self.verticalDrop=100

class WaterPortion():
 X = 0
 Y = 0
 Location="null"

class Waterfall():
 def WaterFlowing(self):
  i=0
  while(True):
   self.water.append(WaterPortion())
   self.bed.TraverseUpperStreamBed(i)
self.bed.TraverseDrop(i)
   self.bed.Pool(i)
   print(i,self.water[i].Location)
   i = i+1
 def __init__(self):
  self.water = []
  self.bed=StreamPath(self.water)
  self.WaterFlowing()
def run():
 W = Waterfall()
```

Figure 1. Python program that defines and implements a simple simulated waterfall

transitions of portions, or droplets, of water are integral to the definition of rivers and waterfalls. Similarly, transitions are integral to some other objects such as stars and systems such as the solar system, but most objects are initially static and demonstrate changes only when they are the subject of Transitionals (e.g., a person starts to run). There are many refinements we could introduce. If it were a stream rather than a river, we could adjust the amount of water accordingly and model the bed in greater detail. We could also model the waterfall if it froze. Each cycle of the while loop could check the current state of the water to make sure it is fluid.[4]

Coordination between molecules and macroscopic effects is a common issue for semantic modeling of fluids. In Figure 1, we model the movements of small *Portions* of

[4] This is a version of two-phase validation we have described elsewhere. Validation could also be addressed by contracts such as those implemented in the Eiffel programming language. For semantic modeling, the contracts could include inheritance and semantic tests.

water. However, an actual portion of fluid would not retain continuity over time. Thus, this is an idealized model in which the *Portions* remain unified. Because we are dealing with a qualitative, or nominal, model we could have simply moved *Portions* across the qualitative states. Rather, to add fidelity to the model, we moved *Portions* across numeric units. This makes it an ordinal model rather than a purely qualitative model.[5] Ideally, extending model resolution would be relatively seamless. Because this is an interpreted Python program, we can pause it as it runs and manually examine the value of any variable. In the future, we could add a graphic control panel to support more flexible ways to interact with the model.

Frame Net Analysis

Frame Net (Allen, 2014; Ruppenhofer et al., 2016) is a linguistic resource based on the theory of frame semantics (Fillmore & Baker, 2009). The key idea is that rather than just activating the meaning of scattered words in a sentence, a broad context is activated. Frame Net identifies and describes approximately 1500 frames. One of those frames is Fluidic_Motion: *A Fluid moves from a Source to a Goal along a Path or within an Area.*[6]

The core Frame Elements of *Fluidic_Motion* are *Fluid, Source, Goal,* and *Path/Area*. Other, non-core Frame Elements include *Configuration,* which specifies parameters such as the volume and speed of the Fluid.

In addition, specific Lexical Entries (i.e., words) are associated with each Frame. *Flowing* is a Lexical Entry associated with the Fluidic_Motion Frame from which it inherits the Frame Elements. Although Frame Net does not give a detailed analysis of Flowing it does provide this definition: *To move with a continual change of place among the constituent particles.*[7] The WaterFlowing routine in Figure 1 embeds the core Frame Elements along with features of this definition. Alternatively, the Fluidic_Motion Frame Elements could have been represented with the familiar functional notation: *Flow (Fluid, Source, Pool, Path)*

In the case of the waterfall, a *Path* is a recursive series of segments with width, length, and slope. Other Parameters could be incorporated by adding the *Configuration* non-core Frame Element. In addition, the flexibility of the code could be enhanced if we defined an abstract class *FluidicMotion*.

In Frame Net, the Lexical Entry for a Waterfall is associated with the Natural_Features Frame: *The Locale is a geographical location as defined by shape. This frame includes natural geographic features, including land/ice forms and bodies of water.*

Potentially, we could develop a microworld terrain model for the surrounding environment by linking the Natural_Features Frame and other Frames with the Fluidic_Motion Frame. Controlled vocabularies such as WordNet and the Getty Art and Architecture Thesaurus (AAT) can also enrich the details of waterfalls as natural features.

A much larger standardized vocabulary should be developed that could support hooks for the interoperability of the frames. Beyond a collection of structured Frame Elements, there could be standardized computational descriptions for the Lexical Units. For some definitions, functionality is as important as structure, and those programmatic definitions could include

[5] Ordinal models are not full quantitative models. Quantitative models might employ massively parallel computation using the Multiscale Object-Oriented Simulation Environment (MOOSE) (Dudani, Bhalla, & Ray, 2014; Fishwick, 1996). For the waterfall, a full quantitative model might be based at the molecular level and also fully model the erosion of the stream bed. Nonetheless, although the simulation can be highly numerical, the description may still be mostly qualitative.

[6] https://framenet2.icsi.berkeley.edu/fnReports/data/frameIndex.xml?frame=Fluidic_motion

[7] https://www.merriam-webster.com/dictionary/flow

that aspect. The dimension of Function in models is discussed later. Moreover, there are several other approaches to semantic roles beyond FrameNet to be explored.

EXECUTABLE SEMANTIC MODEL OF THE CARDIOPULMONARY SYSTEM

Overview

We also developed a stylized, textbook-level object semantic model of the Cardiopulmonary System as a proof-of-concept (Figure 2). This approach implemented entities as objects and used message passing to communicate between the objects. As shown in the figure, O_2-rich air moves from the nose to alveoli in the lungs. There, O_2 diffuses into the blood, and CO_2 from the blood passes back into the lungs. The O_2 then circulates through the heart and out to the body. Cells in the body use the O_2 and return CO_2 into the blood. A sensor in the medulla triggers the diaphragm if there is too much CO_2 in the blood. Our model is asynchronous and has several threads. We implemented two main sub-systems: circulation and respiration.

For the circulatory sub-system, the contraction of the heart spreads across its four chambers. Portions of blood are moved throughout the system with each heartbeat. We modeled capillaries at the lungs, at a typical body cell, and at the sensor in the medulla. For the respiratory sub-system, a portion of O_2-rich air is drawn through the nose into the alveoli in the lungs. The O_2 diffuses into a portion of blood in the capillaries and, at the same time, CO_2 passes from the blood into the portion of air. That portion of air is then exhaled through the nose out to the external air. For the capillaries, we modeled a physical structure (i.e., the capillary tube) as a non-material Site or Place, and a small portion of fluid (either blood or air) contained within that Site or Place. As with the Portions of Water in the Waterfall example, we assume that these Portions pass through the circulatory system intact. This is a sort of Galilean modeling. The modeled blood and air flows follow the solid black lines shown in Figure 2. For both the respiratory and circulatory systems, explicit connection relationships were set between the nodes. Further, the presence of a connection was validated before any Portions were pushed. Because the network never changed in this version of the model, that check was mostly redundant. Rather than trying to synchronize pushing individual portions through the circulatory system, all updates were calculated and then implemented simultaneously. The timing for the modeled motion of the fluids was rough. The respiratory system was triggered independently by the Medulla and the circulatory system by the SA Node.

Figure 2 shows the major the major organs (lungs, heart, and brain). The solid black lines connecting the organs show the path for blood and airflow. The black dotted line shows the Phrenic Nerve which controls the diaphragm. Also shown are enlargements of the capillaries at a typical cell and at a typical alveolus. Respiratory and circulatory sub-systems can be identified along with mechanisms for nerve signaling and gaseous diffusion. While the connectivity is included in the model, relative spatial positions are not.

Figure 3 shows a sample output as the program runs. Because the sub-systems are asynchronous, the updates are interleaved. In the figure, we can see examples where the SA Node triggers and Portions of blood are pushed in the circulatory sub-system, cases in which diffusion occurred, and an example of a contraction of the diaphragm causing a breath to be taken.

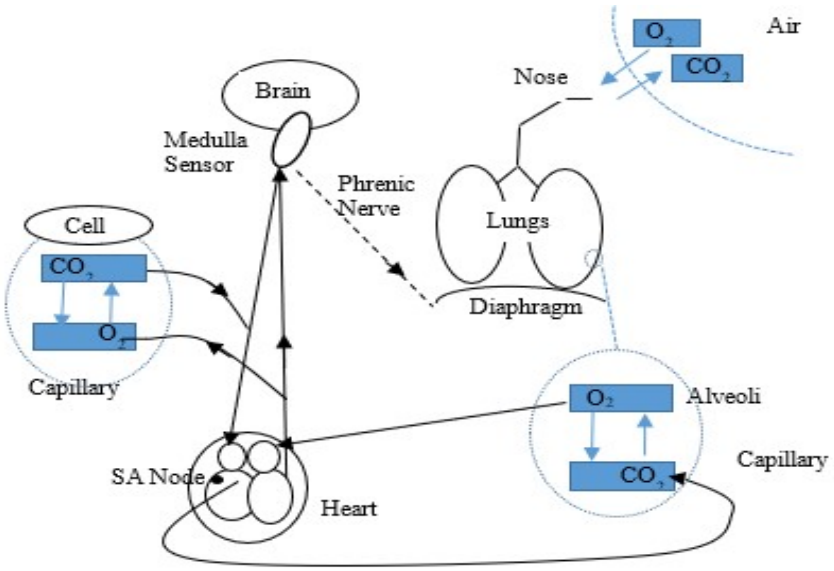

Figure 2. Schematic model of the cardiopulmonary system

```
mixing external air
diffusion check
AlvCapBlood O2 diffusion
CellCapBlood O2 diffusion
pushed LeftAtriumBlood
pushed LeftVentricleBlood
pushed MedullaCapBlood
pushed cellCapBlood
pushed RightAtriumBlood
pushed RightVentricleBlood
pushed AlvCapBlood
trigger updates
diffusion check
CellCapBlood O2 diffusion
mixing external air
```

```
CellCapBlood O2 diffusion
pushed LeftAtriumBlood
pushed LeftVentricleBlood
pushed MedullaCapBlood
pushed cellCapBlood
pushed RightAtriumBlood
pushed RightVentricleBlood
pushed AlvCapBlood
trigger updates
SAnode pulse
inhale cycle
past phrenicNerve trigger
into diaphram contract
completed inhale ExternalAir to Nose Air
completed inhale Nose Air to Alv Air
```

Figure 3. Two panels showing section of continuous output from the cardiopulmonary semantic simulation

Commentary on the Cardiopulmonary System Example

This model implements systems, sub-systems, and mechanisms. The fluid motion of the air and the blood could be implemented with the *FluidicMotion* abstract class described in the earlier for the waterfall. The model incorporates many simplifications; it uses only simple motion and simple structures.

The nerve impulse was not modeled in detail but could have been implemented as sodium channels opening and closing along the neuron. The sensors can be seen as providing information about the CO_2 level, which is transmitted to the diaphragm.[8]

We could generate a family of related models of different levels of complexity. On the one hand, we could model cell metabolism. On the other hand, we could extend this model to include interaction with other body systems such as the digestive system. Moreover, the

[8] The mechanism for transmitting the nerve signal is relatively straightforward. The notion of "information" may be most useful as a shortcut at the model-level when the mechanisms are not specified.

model of the entire cardiopulmonary system as described here could be adapted for use across species of mammals.

PARTS

Functional and Structural Parts

We focus here on Functional Parts, which are Objects that are essential for a state change in the models. Mechanisms link behavior and parts; they show how the functionality is accomplished. We also allow Structural Parts such as the stream bed in the earlier section.

Structural Parts are essential for the physical instantiation of a Mechanism without contributing directly to the functionality.[9] Structural Parts support the interaction of parts but do not show state changes themselves. For instance, the bracket holding a car's carburetor in place would be a structural part supporting the carburetor. The bracket may be composed of several separate pieces and the array may be considered an Assembly.[10]

Several fields have developed resources for structural and functional parts that could be incorporated into our models. For instance, mechanical engineers and computer graphics researchers have developed CSG (Constructive Solid Geometry)[11] which is a structural modeling framework that allows for the relative motion of parts such as the motion of joints. Similarly, frameworks from anatomy (e.g., Bard, 2008; Grizzi, & Chiriva-Internati, 2005) and structural biology could be incorporated.

Composition and Portions of Matter

Beyond the usual notion of Functional Parts (e.g., the engine, tires, and body of a car), we sometimes consider the materials of which an Object is composed. We saw examples in the earlier sections where water, blood, and air are better described as Substances rather than Objects. (Keet, 2016) describes a typography of Substances and Mixtures. Also in both of these earlier sections, we defined Portions of the Substances and took the Portions as Objects (Vogt, 2019). Some of the challenges in this area can be traced to the uneven treatment of atoms and molecules as Objects. Adding or subtracting a few molecules of water does not materially change a lake. The difference between water molecules and the lake is a granular perspective. We propose a multi-granular perspective which allows modeling the different levels at the same time.

STATES, TRANSITIONS, AND BEHAVIOR

Objects change; but how should we model those changes? We adopt State Transitions. The notion of State is widely recognized in information processing. Roughly, a State is a distinct condition of an object for a given period. We distinguish between Object-Centered States, which are associated with a single Object, and Ensemble States, which describe the interaction of several independent Objects.

---

[9] Paoletti (2018) proposes that the Structure of an Object consists of all the internal relationships of that Object. Further, he proposes that Structure should be considered as an entity in its own right. This seems consistent with the definition of Objects by classes in an object-oriented programming language but differs somewhat from the use of Structure in this section.

[10] Although such a bracket can be said to provide a function -- that of positioning the carburetor to allow it to interact with other parts –that is a secondary function.

[11] https://en.wikipedia.org/wiki/Constructive_solid_geometry

Object-Centered States

The State of an Object is interwoven with its Parts. In many cases, it is the activity of Parts that makes a State. One example is that when a person is running their legs are moving quickly back and forth. Another example would be describing phase changes in a material which is due to the excitation of its atoms/molecules. State changes for a traffic light (cf. Allen & Jones, 2018) can be modeled as State changes of separate (green, yellow, red) lamps within the light. But, ultimately the State of those lamps is due to the activity of electricity (i.e. electrons) flowing. The example of the Waterfall is unusual because the flowing water is not a State but is integral to it being a waterfall. We refer to the traffic light and the waterfall as having multi-granular states because the parts (e.g., the electrons) are at a different level than the parent object. Other State changes are due to changes in functionality (e.g., usable/unusable) although these differences may, ultimately, also be based on changes of Parts. Still other types of State changes are due to changes in space and/or time.

Ensemble Interaction and States

We can model the States of interacting Objects. An underlying question is when independent Objects may better be considered as a single, unified Object. Examples include one and the clothes one is wearing, a couple who gets married, or a car that gets a new coat of paint. These might be implemented as Relational Qualities connecting two Objects (e.g., Arp, Smith, & Spear, 2015, pp 97-98) though it would be helpful to have more nuanced descriptions of the possible types of relationships that could be used. In some cases, we can treat the new, unified object as a replacement for the previous object. The car with a new coat of paint could be considered as an update to the car. In other cases, new entities exist but are highly transient such as a party with its revelers or a chemical bond existing fleetingly during a reaction. In still other cases, these Objects retain a context-dependent duality. When a person becomes part of a couple, that person's professional activities may be relatively unchanged while other activities are as part of the couple. This is comparable to the relationship between molecules and the atoms that compose them. In yet other cases, the Objects retain their distinct identities but are joined by their participation in an activity. We consider a pianist as distinct from the piano that is being played although both are involved in the same scenario.

Transitionals and Behavior

States imply the possibility of State changes. We use the more general term Transitionals to include other types of transformations of objects such as birth, death, splits, and merges. State changes are often changes of parts, but not all changes of parts result in State changes of the parent Object or vice versa. Thus, we can distinguish Behavior from State changes. When a person is running, their legs are constantly moving but are not constantly changing State. Likewise, water molecules are always vibrating and spinning but small changes in those activities do not change the State of the water.

In some cases, we might ask whether an Object has States or even Behavior. For example, some physical objects such as a bridge or chair provide a function (i.e., a river crossing or a seat), but have little apparent motion. Nonetheless, they do deform when used, and, presumably, the bonds between their atoms and molecules are affected. At a low level of granularity, they are in a state of tension when providing a function.

Relationship to Object-Oriented Analysis and Modeling

In earlier papers, we have noted the similarity of semantic modeling to object-oriented analysis and modeling. There is not unanimity about the definition of object-oriented modeling, but we can consider several of the most common features associated with it - abstraction, inheritance, encapsulation, and polymorphism (e.g., Morris, Evans, et al., 1996).

An example of abstraction is our use of abstract methods to describe the contraction of the heart or diaphragm. Objects in our models exhibit inheritance; for instance, different mammals could inherit a general cardiopulmonary system model such as described earlier.[12] Encapsulation suggests that the methods used by an Object should be hidden and inaccessible except by hooks at the top level. However, because we have multi-granular modeling as in the traffic light and waterfall example in the above section on *Object-Centered States*, the Parts of Objects and the methods associated with them are generally exposed Polymorphism means that methods may be overloaded. While our examples do not include Polymorphism, potentially they could. Object-oriented languages such as Smalltalk propose that objects should communicate via message passing. In the cardiopulmonary example, the Phrenic Nerve can be thought of as passing messages to the diaphragm. For models of physical interaction with collisions between objects, those collisions can be considered as a type of message passing.

FUNCTIONS, CAUSATION, MECHANISMS, SYSTEMS, AND MICROWORLD

Function

Mechanical engineers often identify three dimensions for describing their models: Function, Behavior, and Structure (FBS or SBF) (Goel, Rugaber, & Vattam, 2009). In PARTS above, we addressed Structure such as Parts and Assemblies; and in STATES, TRANSITIONS, AND BEHAVIOR above, we addressed Behavior (and State). In this section, we turn to Function. We include how something is used along with its function.

To assert that an Object has a Function we need to know its broader context. If someone in the Middle Ages had, somehow, created an object identical to a modern carburetor, we would not say that that object has the function of a carburetor. In other words, we assert that a Function is relative to a Mechanism, System, or Scenario, and a description of a Function needs to include a context—perhaps by identifying a Mechanism or System with which it is associated.[13] For example, while the study of anatomy is most often concerned with structure, reconciling parts and functions is a traditional issue for anatomy (Bard, 2008; Mungall, Torniai, Gkoutos, Lewis, & Haendel, 2012; Rosse & Meijino, 2008).

Causation

Modeling sequences of events assumes that the events do not happen at random, that they are, in some sense, based on earlier events. We say that events are caused by those earlier events. Nonetheless, the notion of causation is highly contentious. Much of the problem is due to the difficulty of ascertaining causation retrospectively. That often involves possibly unreliable

[12] It is unlikely that simple single inheritance can always be applied across entire complex objects and systems. Presumably, some way of indicating exceptions and inheritance by Function/Use or Behavior could be adopted.

[13] Perhaps the need for the existence of a Mechanism or System is what the Basic Formal Ontology (BFO) intends by declaring that Functions are Realizables. For example, (Arp, Smith, & Spear, 2015, p 104) states that "to detoxify its containing organism is a function of this liver". However, BFO has no clear notion of Mechanism, System, or Scenario.

evidence and uncertain inference. Similarly, assessing causation is difficult when the Objects, Parts, and Transitions are ambiguous. However, in a model, where we define all of the Objects, Parts, and Transitions, there is little controversy about causation.

Basic science identifies types of Objects in the world as well as their Parts and Transitions. In other words, basic science can be seen as developing consistent descriptive frameworks (Arp, Smith, & Spear, 2015, p. 12-13). By comparison, applied science attempts to use those frameworks to predict or explain real-world phenomena. Given science's success, it is reasonable to identify objects as entities with "causal powers" (Paoletti, 2018) and that what is real is what is consistent with scientific results.

Mechanisms

Allen (2018) described some simple semantic Mechanisms based on Petri Nets.[14] Mechanisms based on Petri Nets are the foundation of functional descriptions. Although we can trace a mechanism between any two causally connected Objects, we most often focus on Mechanisms that are responsible for the functioning of a system. In some cases, such as the Central Dogma in biology, which describes the steps from the transcription of DNA to the production of proteins, a Mechanism may be multi-granular. That is, a transition at a low level can dramatically affect the dynamics of a system at a higher level across time.

A side effect is a change that is not directly needed for the completion of a given Mechanism. For instance, a reaction may produce heat that does not affect the completion of the Mechanism that produced it. But that heat may affect other Mechanisms. Although developers are urged to avoid them in well-designed systems, side effects are inherent in many natural systems and must be included in models.

Systems, Sub-Systems, Microworlds, and Scenarios

A system is a complex object which is composed of several interacting Mechanisms (cf., Allen, 2014). Several different types of systems may be identified. Many are purely feed-forward, while others have feedback. Some of the latter have an internal regulator and manage homeostasis while others of them are chaotic even if they attempt regulation. We can model much of the behavior of systems with feedback with semantic tools, although for purely semantic models the non-linearities need to be described qualitatively.

"A Microworld is a restricted, idealized model of the world containing only those relations and entities of interest in the particular reasoning system being designed" (Davis, 1990, p 6). A System implemented in a Microworld may show spatial relationships (e.g. *next to*) and may reflect other constraints (e.g., *is symmetrical*). We also allow ambient properties such as temperature, humidity, and even gravity.[15]

To distinguish the Microworld as a frame or platform from its representation of the world, we call the latter Scenarios. We have considered Scenarios at several different points in this work. A terrain model incorporating the Waterfall example and the Cardiopulmonary system in Figure 2 are Scenarios. We might use a variation of Figure 2 to describe what happens when a person is running or what happens when a person's heart stops beating. This is not intended to be generalized inference about broken mechanisms); rather, it is a targeted extension of a model to handle a specific situation. It can be considered as a semantic modeling alternative to referential ontologies (Ortmann & Daniel, 2011).

[14] In comparison to the model of the cardiopulmonary system above, the models in (Allen, 2018) were ad hoc functional accounts without systematic descriptions of Behavior or Structure.
[15] Standard temperature and pressure (STP) is the default.

MODELING

Model Operators

In addition to the data types such as Objects, Transitionals, and Mechanisms, we also allow familiar data modeling features such as cardinality. Thus, we may specify that a mammal has two ears, two eyes, and two or four legs. Because the programming language is integral to the modeling, we also need to recognize language features such as looping, conditionals, concurrency, and threading.

While the current models are primarily qualitative (i.e., categorical or nominal), they also include simple binary comparisons (e.g., high and low concentrations of O_2 and CO_2). They could be extended across richer levels recognized by data analysis as ordinal, interval, and ratio models.

Meta Operators and Annotation

Meta-operators are elements incorporated into a model to make it more tractable. Along with Annotations, they show how the model does not fully reflect reality. The meta operators would include approximations to continuous operations (e.g., diffusion), cross-granular descriptions (Mulkar-Mehta, Hobbs, & Hovy, 2011), approximations to stochastic processes, picking typical or random examples, and compensation for non-linearities. Structured annotations may include describing simplifications (i.e., eliminating unnecessary detail) and idealizations (i.e., intentionally inaccurate statements which, nonetheless, improve clarity).[16] Other types of annotations could include comments from users.

Explanations and User Interaction

As described in the earlier section on *Commentary about the Waterfall Example* in STRUCTURED DEFINITIONS, a graphical control panel could be developed for user interaction with the programs. The interface would have modes suitable for different types of users (e.g., students, researchers, editors). In addition to graphical interaction, there could be a natural language interface. That could employ text generation based on the model operators and annotations described in the previous sections. The text generation could also apply discourse elements (Allen, 2013) such as those from Rhetorical Structure Theory (RST) (Mann & Thompson, 1988). It might even support tutoring. We may also be able to develop a grammar for manipulating the model. Such a grammar could be a sort of interactive, model-based programming language.

DISCUSSION

Validation and Extrapolating Mechanisms

While our primary goal is description, it is useful to validate the programs as much as possible. For instance, we should validate that the application of Transitionals does not violate any of the assertions made by relationships. The example in the earlier STRUCTURED DEFINITIONS section includes some checking, but more is needed. This can be done with two-phase validation which was described in our previous work (Allen & Chu, 2015). That is, applying traditional validation of triples after every step of the model.

[16] See https://plato.stanford.edu/entries/models-science/

Semantic Modeling for Scholarly Communication

We expect that models like those described here will be the basis of direct representation research reports (Allen, 2017). Such reports would describe scientific research based on a structured knowledge base. In addition to modeling natural phenomena, the models would incorporate structured descriptions of research workflows and data analysis workflows (Allen, 2018). Claims could be made about aspects of Mechanisms and evidence from observations applied to support or contradict those claims. Deemphasizing text will eventually lead to less need for text mining. These techniques can also be applied to developing models of communities covered by historical newspapers (Allen, 2014; Barker, 1978).

Potentially, both students and researchers would benefit from browsing knowledge structures. Standards for constructing the models could be developed and libraries of models could be collected. If we develop libraries of composable mechanisms[17] and models, then users should be able to bridge out to related models and also to drill down from more general models to more detailed models. Moreover, broken Objects and Transitions will likely affect the Mechanisms in which they participate (e.g., Rector, 2008; Rosse, & Meijino, 2008; Schonfeld, Rozell, & Gkoutos, 2008; Thaagard, 2001). A well-documented library of Mechanisms would be of great value for making predictions about the impact of and solutions to those broken Objects and Transitions.

Envoi

We have sketched a broad framework and emphasized a broad synthesis of many approaches. In the STRUCTURED DEFINITION section, we developed a programmatic implementation of natural language definitions which is consistent with Frame Net. In the EXECUTABLE SEMANTIC MODEL OF THE CARDIOPULMONARY SYSTEM section, we implemented a semantic simulation of interacting components of the cardiopulmonary system. From the section on PARTS to the section on MODELING, we considered issues for the implementation. Although these examples are relatively simple, they are potentially composable and scalable. We envision that the current generation of ontologies will eventually be replaced by terms with fully structured definitions. However, challenges remain, such as how to manage consistency in the assumptions made when describing the interaction of several related mechanisms.

REFERENCES

Allen, R.B. (2013). Model-oriented information organization: Part 2, discourse relationships. *D-Lib Magazine, 19*(7), 5. https://doi.org/10.1045/july2013-allen-pt2

Allen, R.B. (2014). Frame-based models of communities and their history. In A. Nadamoto, A. Jatowt, A. Wierzbicki, & J.L. Leidner (Eds.), *SocInfo 2013 Workshops: QMC and Histoinformatics 2013, 25 November 2013, Kyoto, Japan* (LNCS 8359, pp. 110-119). Springer.

Allen, R.B. (2017). Rich semantic models and knowledgebases for highly-structured scientific communication. *arXiv preprint, arXiv:1708.08423.*

Allen, R.B. (2018). Issues for using semantic modeling to represent mechanisms. *arXiv preprint, ArXiv:1812.11431.*

[17] For example, the introduction to many research papers starts by listing known mechanisms associated with the phenomenon under study.

Allen, R.B., & Chu, Y.M. (2015). Architectures for complex semantic models. In *2015 International Conference on Big Data and Smart Computing (BIGCOMP), 9-11 February 2015, Jeju, South Korea* (pp. 254-261). IEEE Explorer. https://doi.org/10.1109/35021bigcomp.2015.7072809

Allen, R.B., & Kim, Y. (2018). Semantic modeling with foundries. *arXiv preprint, arXiv:1801.00725.*

Allen, R.B., & Jones, T.K. (2018). XFO: Toward programming rich semantic models. *arXiv preprint, arXiv:1805.11050.*

Arp, R., Smith, B., & Spear, A.D. (2015). *Building ontologies with basic formal ontology*. The MIT Press.

Bard, J. (2008). Anatomical ontologies for model organisms. In A. Burger, D. Davidson, & R. Baldock (Eds.), *Anatomy ontology for bioinformatics: Principles and practice* (pp. 43-57). Springer,

Barker, R.G. (1978). *Habitats, environments, and human behavior: Studies in ecological psychology and eco-behavioral science from the Midwest Psychological Field Station, 1947-1972*. Jossey-Bass Publisher.

Dudani, N., Bhalla, U.S., & Ray, S. (2014). MOOSE, the multiscale object-oriented simulation environment. In D. Jaeger, & R. Jung (Eds), *Encyclopedia of computational neuroscience*. https://doi.org/10.1007/978-1-4614-7320-6_257-1

Davis, E. (1990). *Representations of commonsense knowledge*. Morgan Kaufman Publisher.

Fillmore, C.J., & Baker, C. (2009). A frames approach to semantic analysis. In Heine B. & Narrog, H. (Eds.), *The Oxford handbook of linguistic analysis*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199544004.013.0013

Fishwick, P.A. (1996). *Extending object-oriented design for physical modeling*. CiteSeer. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.8429

Galton, A., & Mizoguchi, R. (2009). The water falls but the waterfall does not fall*: New* perspectives on objects, processes and events. *Applied Ontology*, *4*(2), 71-107. https://doi.org/10.3233/ao-2009-0067

Goel, A., Rugaber, S., & Vattam, S. (2009). Structure, behavior and function of complex systems: The structure, behavior, and function modeling language. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *23*(1), 23-35. https://doi.org/10.1017/s0890060409000080

Grizzi, F., & Chiriva-Internati, M. (2005). The complexity of anatomical systems. *Theoretical Biology and Medical Modeling*, *2*(1), 26. https://doi.org/10.1186/1742-4682-2-26

Hornsby, K.S., & Cole, S. (2007). Modeling moving geospatial objects from an event-based perspective. *Transactions in GIS*, *11*(4), 555-573. http://doi.org/ 10.1111/j.1467-9671.2007.01060.x

Keet, C.M. (2016). Relating some stuff to other stuff. In Proceedings of *Knowledge Engineering and Knowledge Management (EKAW 2016), 19-23 November, Bologna Italy* (pp. 368-383). Springer. https://doi.org/10.1007/978-3-319-49004-5_24

Mann, W.C., & Thompson, S.A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text: Interdisciplinary Journal for the Study of Discourse*, *8*(3), 243-281. http://doi.org/10.1515/text.1.1988.8.3.243

Morris, D., Evans, D., Green, P., & Theaker, C. (1996). *Object oriented computer systems engineering.* Springer. https://doi.org/10.1007/978-1-4471-1015-6

Mulkar-Mehta, R., Hobbs, J., & Hovy, E. (2011). Granularity in natural language discourse.In J. Bos & S. Pulman (Eds.), *Proceedings of the Ninth International Conference on*

Computational Semantics, 12-14 January 2011, Oxford, UK (pp. 360-364). https://www.aclweb.org/anthology/W11-0143

Mungall, C.J., Torniai, C., Gkoutos, G., Lewis, S. E., & Haendel, M.A. (2012). Uberon, an integrative multi-species anatomy ontology. *Genome Biology, 13*(1), R5. http://doi.org/10.1186/gb-2012-13-1-r5

Ortmann J., & Daniel D. (2011). An ontology design pattern for referential qualities. In A. Lora, W. Chris, A. Harith, T. Jamie, B. Abraham, K. Lalana, ... B. Eva (Eds.), *The Semantic Web, 10th International Semantic Web Conference (ISWC 2011), 23-27 October, Bonn, Germany* (LNCS 7031, pp. 537-552). Springer. https://doi.org/10.1007/978-3-642-25073-6_34

Paoletti, M.P. (2018). Structures as relations. *Synthese*, 1-20. https://doi.org/10.1007/S11229-018-01918-8

Rector, A. (2008). Anatomy for clinical terminology, In A. Burger, D. Davidson, & R. Baldock (Eds.), *Anatomy ontology for bioinformatics: Principles and practice* (pp. 43-57). Springer.

Rosse, C., & Meijino, J.L.V. (2008). The foundation model of anatomy ontology. In A. Burger, D. Davidson, & R. Baldock (Eds.), *Anatomy ontology for bioinformatics: principles and practice* (pp. 59-117). Springer.

Rovetto, R.J., & Mizoguchi, R. (2015). Causality and the ontology of disease. *Applied Ontology, 10*(2), 79-105. http://doi.org/10.3233/ao-150147

Ruppenhofer, J., Ellsworth, M., Petruck, M.R.L., Johnson, C.R.J., Baker, C.F., & Scheffczyk, J. (2016). *Framenet II: Extended theory and practice* (Rev.). https://framenet2.icsi.berkeley.edu/docs/r1.7/book.pdf

Schonfeld, P.N., Rozell, B., & Gkoutos, G.V. (2008). Toward a disease ontology. In Burger, D. Davison, & R. Baldock (Eds.), *Anatomy ontology for bioinformatics: principles and practice* (pp. 119-130). Springer.

Thagard, P. (2001). *How scientists explain disease*. Princeton University Press.

Vogt, L. (2019). Levels and building blocks—toward a domain granularity framework for the life sciences. *Journal of Biomedical Semantics, 10*(4). http://doi.org/10.1186/s13326-019-0196-2