From Whiteboard to Model: A Preliminary Analysis

Praveen Paritosh[1] and Will Bridewell[2]

[1] Qualitative Reasoning Group, Northwestern University, Evanston, IL, USA

[2] Computational Learning Laboratory, Center for the Study of Language and Information, Stanford University, Stanford, CA 94305 USA

{paritosh@cs.northwestern.edu, willb@csli.stanford.edu}

Abstract

Building models of a complex system such as an ecosystem or a chemical plant is an arduous task that can take several person months to complete. One rarely knows the scope of the model, its assumptions and claims, at the outset of the task, let alone how to state those in a formal language. To make this task manageable, modelers start at the whiteboard – by making free-form drawings that capture their current understanding of the studied system. These drawings need not conform to any particular ontology and may lack internal coherency or consistency. Nevertheless, such drawings can help organize one's thoughts and can capture key participants and relationships in the dynamic system. We argue that these free-form drawings facilitate the modeling process, based on evidence from modeling in practice. We analyze the relationship between free-form drawings and formally encoded models. We then suggest how to exploit these relationships to develop a modeling environment that supports a tighter integration between conceptual and detailed modeling.

1 Introduction

Model building is a common and vital task in the sciences. The resulting artifacts of thought let us test the implications of our theories, help us better understand complex systems, and support effective communication and education. Moreover, models specified in a computable language support prediction and diagnosis, thereby enabling discussions about things such as the migration of killer bees and the efficacy of carbon sequestration. Unlike theories, models describe specific situations, real or imagined. For example, the theory might discuss the migration of a general population across a landmass, whereas a model would make specific claims about the movement of *Apis mellifera scutellata* into south Texas. Given this model, one might predict that residents of El Paso will be knee deep in bee-riddled corpses by March 1979. In comparison, hypotheses are singular, testable statements such as "building a border fence 50 meters high will protect us all from the bee invasion." With an appropriate model, one could gauge the plausibility of this hypothesis before committing the required resources.

Despite the ubiquity of modeling, there have been relatively few analyses of the task of modeling. More often, researchers emphasize the effectiveness of an encoding tool [e.g., Bridewell et al., 2006] or the use of the resulting model. Recent work in chemical engineering [Foss, Lohmann and Marquardt, 1998] and in educational settings [Sins, Savelsbergh and van Joolingen, 2005] exploring the process of modeling itself are more the exception than the rule. One general finding is that the modelers benefit from multiple representations and that each one has its own merits [Lohner, van Joolingen, and Savelsbergh, 2003]. These results suggest that modeling environments should, in principle, support different views of the artifact and that these views should map onto each other.

In the qualitative reasoning community, Bredeweg and colleagues have recently described a framework for building qualitative models [Bredeweg et al., in press]. One of their findings is that the use of loosely constrained conceptual models provided considerable, if not necessary, support for the development of formal qualitative models. Bredeweg used a concept map to capture the earlier stage of modeling. We believe that the simplicity and popularity of concept maps make them a suitable interface for describing the pre-formal free-form modeling that happens at the whiteboard, and the rest of this paper will stick to this assumption[1].

The natural extension of these findings is the development of a modeling environment that provides a tighter integration between building the concept map and the detailed model. In this paper, we describe how such a system might be designed. The next section discusses the modeling process in more detail, with special emphasis on the utility of pre-formal conceptual models. We then describe how concept maps facilitate building detailed models. Next we present a catalog of ontological relationships and mapping operations between elements in the concept map and the detailed model. Finally, we conclude with questions and future research issues that this discussion generates.

[1] This is a simplifying assumption, the whiteboard provides a much richer interface, perhaps more like the sketching systems, e.g., sKEA [Forbus and Usher, 2002].

2 The Modeling Process

Model building forms a part of larger tasks such as design and scientific investigation. The task goals influence the trade-offs among generality, realism, and precision of the model [Levins 1966]. Models built for communicating the relationships in a complex system tend to be more general, while models built for process control emphasize precision, and so on. One's available knowledge and data also influence the modeling task. Domains such as chemical engineering and circuit design are knowledge-rich, which enables a realistic expression of the entities and relationships within the modeled system. Other domains, such as ecology, are less theory-driven, and the amount and type of data will influence one's modeling decisions. Models range from being purely descriptive and explanatory, e.g., in political and social domains to being predictive, e.g., in engineering.

Foss and colleagues [1998] performed a field study of the modeling process in the domain of chemical engineering, in which they interviewed sixteen modeling practitioners with an average modeling experience of over ten years. These interviews followed a case study wherein the modelers described a realistic modeling experience. On the basis of these interviews, Foss et al. identified six distinct activities: 1) problem understanding, specification, initial data collection, 2) conceptual modeling and model representation, 3) implementation and verification, 4) initialization and debugging, 5) validation and 6) documentation. Notably, the modeling process is not linear. That is, the modeler may freely move among these six activities without any fixed pattern. However, as model refinement progresses, the modeler moves through the chunks sequentially as a moving window capturing more than one chunk at a time. As the modeling process goes by, the degree of back-steppings to the earlier chunks diminishes in favor of forward-steppings to the later chunks. Nevertheless, there exist numerous iterations between the chunks rendering a highly intertwined and complex modeling process.

Foss's study offers important insights for builders of modeling environments. First, modeling is not a strictly progressive refinement from conceptual to detailed models. This finding suggests that environments should support links among the tasks so that modelers need not shift to external media as they work. Second, the modeling environment must provide tight integration between the various modeling activities. Being able to work with several representations of a model becomes problematic when they are unsynchronized. The environment should treat each representation as an index into the others so that the modeler can move about freely with ease. And third, no matter the richness of knowledge or data about an environment, conceptual modeling remains important. Thus, builders of modeling tools should consider including various level of representation – including those that permit inconsistency as is inevitable as one begins to model, into their software.

One of the first stages in modeling a complex system involves the identification of the model's scope, which includes the relative entities and relationships expressed at a high level. This task fits well into Rittel and Webber's (1973) notion of a *wicked problem*. In particular, the problem definition is usually vague and evolving, proposed

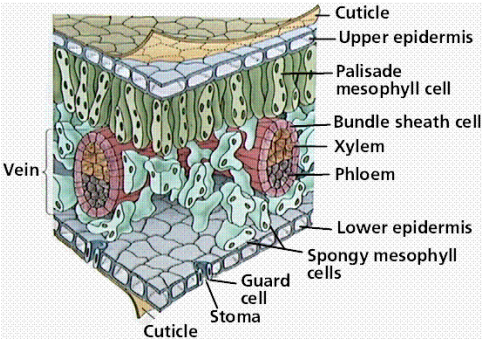

Figure 1. A visual diagram of the cross section of a leaf, reprinted from Farabee [2001]

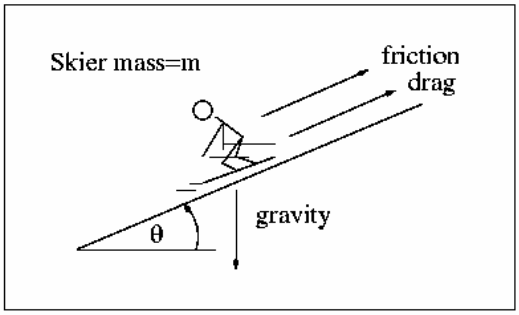

Figure 2. A spatial free-body diagram of a skier.

solutions can create new problems, and multiple solutions may exist with no obvious measures of preference. To begin, one often represents the target system with free-form text descriptions and drawings on paper or whiteboard. As with the entire modeling process, the goal is to make one's knowledge explicit, but at this stage issues of syntax and semantics can serve as barriers and interfere with one's creativity. So, when working with pen and paper one introduces objects and relationships without concern for incomplete specifications, consistent typology, or formal correctness.

Most of the quantitative modeling environments today (e.g., AspenTech's HYSYS, ASCEND, SPEEDUP, STELLA, Simulink), primarily focus on the formal encoding of models without much support for the free-form conceptual modeling that takes place on a whiteboard. On the other hand, qualitative modeling environments like VModel [Forbus *et al.*, 2004], Betty's Brain [Leelawong 2005], Garp[2], among others, provide richer support for less detailed models. However, these are

[2] Downloadable from http://hcs.science.uva.nl/QRM/software/

not unconstrained enough for capturing the possible inconsistency and ambiguity of the whiteboard drawing. Underlying each of these environments is a modeling ontology that constrains and restricts what can be drawn, which is precisely what gives these environments power to reason with the models built using them. There are a large number of software tools available as "mind-mapping tools" [Buzan 1991] that support pre-formal unconstrained drawing. The end result, however, in mind mapping is the drawing that is produced. There is very little work on elaborating or fleshing the output of mind map into a model that can be reasoned with.

Lets look at the different types of drawings that are built while modeling. We classify free-form diagrams into the following three categories:

1. *Visual drawings* are faithful to the salient spatial relationships and bear apparent resemblance to the object or system being drawn. Figure 1 shows a visual diagram of a leaf.
2. *Spatial drawings* use the spatial layout of the drawing medium. Examples include course of action diagrams and free body diagrams in classical mechanics. In this representation, one introduces abstractions and metaphorical conventions such as arrows that convey spatial direction. Figure 2 shows a free body diagram of a skier.
3. *Abstract drawings*, such as UML diagrams, organizational charts, and concept maps ignore the implicit spatial dimension of the drawing medium. In these figures, the relative location of two objects does not necessarily communicate a real spatial relationship. Figure 3 shows an abstract drawing, a concept map.

We admit that free-form diagrams are often complex and rich with implicit knowledge. An aspect of complexity of free-form diagrams is that they can contain different parts that are visual, spatial and abstract in the same diagram, and humans are able to rely on vast commonsense knowledge to interpret it. Ideally, we would like to provide the modeler the freedom of drawing on the whiteboard, but given the complexity of automatically understanding them, we restrict ourselves to the third type above, abstract diagrams. One possibility is to take the sKEA approach [Forbus and Usher, 2002], and allow the modeler to explicitly label every element of the drawing using an ontology like the Cyc[3] knowledge base.

We believe that concept maps are attractive for the abstract diagrams for their simplicity and flexibility. Concept maps [Novak and Cañas, 2006] are graphical tools for organizing and representing knowledge. The power of concept maps comes from the simplicity of the ontology: box-and-lines. Boxes denote concepts and have linguistic labels that identify what they represent, and lines specify a relationship (causal, spatial, etc.) between two concepts. Propositions contain two or more concepts connected using linking words or phrases to form a meaningful statement. Recently, Bredeweg *et al.*, [2006] included support for concept maps in the Garp3 system in the form of a *sketch mode*. However, in their software, the elements of the sketch are not connected to the elements of the detailed model. Here, we emphasize the value of connecting these representations.

The ease of concept maps comes at a price. First, one cannot simulate concept maps or use them to make strong predictions about system behavior. Second, one may explain away phenomena by leaving out important, nontrivial details. For example, a concept map that claims "carbon sequestration *reduces* global warming" might be too simplistic and explain away the complex mechanisms of the process. Put simply, it is possible to make models that state the very fact that the model ought to explain or predict, without providing any richer explanation. And third, one may assume shared understanding of linguistic labels, which can hide the one's preconceptions behind the ambiguity of meaning. The use of a formal, shared vocabulary, such as Cyc[1] for naming the concepts and relationships can safeguard against this problem to a large extent, but at the cost of representational freedom.

The modeling process consists of fleshing-out the concept map to a more detailed model. We call the shift to a more formal representation (i.e., one that can be reasoned over) *encoding.* This step involves moving to a well-defined ontology, such as Forrester diagrams [Forrester, 1961], qualitative process theory [Forbus, 1984], or mathematical equations, and assumes a firm understanding of the concepts. Beginning at the formal stage can be somewhat challenging, but the concept map constrains what one will encode and facilitates the formalization procedure. In the next section, we describe this relationship in more detail.

3 Concept Maps Facilitate Modeling

As described above, the concept map identifies the entities and relationships that need to be further encoded and elaborated in the detailed model. Introductory texts on modeling in various domains, e.g., biological systems [Haefner, 2005], ecological modeling [Jorgensen, 2001] advise modelers to begin with such a drawing of the system. This points to the first benefit of concept maps: ease of knowledge elicitation. Knowledge elicitation is facilitated as the concept map allows the expert to express their mental model in a vocabulary that is close to their models by allowing linguistic labels for entities and relationships. Furthermore, the concept map makes is easier to try out ideas and cast them aside if they fail to satisfy the modeling goals and constraints. In his landmark book, Productive Thinking, based on a case study of Albert Einstein, Wertheimer (1945) argues that a bottleneck to scientific breakthrough is overcoming the structure of existing theories. By providing a freer ontology, concept maps might make it easier for this to happen. In the

[3] http://www.cyc.com/

NatureNet Redime[4] effort to build qualitative models of ecological systems, a first step has been building a textual description and a concept map of the system of concern. This claim of ease of knowledge elicitation has indirect support from practical modeling efforts and conventional modeling wisdom.

Second, the concept map is an important aspect of documentation of the modeling process itself. It captures the conceptual evolution of the modelers' thought process. It also presents a higher level description of the detailed model, in the sense of requirements in software [Jackson, 1995] and design rationale [Moral and Carroll, 1996]. The concept map has communicative value, as it might be easier to get started with the concept map before looking at the simulatable model. For example, in the CMEX[5] project, which was NASA's outreach effort to explain the Mars exploration enterprise to lay people; a collection of about one hundred concept maps detailing various aspects of Mars exploration were released.

Third, for large models that don't fit on a screen, the concept map can be used as a navigational interface for browsing the detailed model by pointing to parts of it that one is interested in exploring in more detail. Furthermore, concept maps contain enough structural information that they can be used to retrieve analogous models from a library of previous models and making analogical suggestions during modeling [e.g., Leake *et al.*, 2003].

4 Usage Scenarios

Designers of model development environments can take advantage of the relationship between concept maps and models both to create a simplified user interface and to scaffold the encoding of formal models. To address the first point, the conceptual model serves as an index to the components of the detailed version, letting one navigate quickly to the relevant sections of the model and access associated interface elements with ease. For the second point, the conceptual model can highlight incompletely specified regions of the system and help the user avoid errors in consistency. In the remainder of this section, we discuss how the conceptual and detailed modeling activities fit in the modeling environment.

There are two possible scenarios of how the modeling environment might support both conceptual and detailed modeling:

1) *Sequential encoding:* One starts with a concept map that is progressively encoded into a simulatable model. In this scheme, the concept map eventually "disappears."
2) *Parallel encoding:* Both the concept and the model are maintained at all times as the modeler goes back and forth elaborating and drawing connections between them.

We believe that the parallel encoding is a more natural model of the modeling task. The Foss *et al.* [1998] study provides direct support of this intertwined nature of modeling activity where one is going back and forth between conceptual and detailed representations. Furthermore, this view suggests that a concept map is more than a stepping stone to a model. It is a continuously developing high-level representation of the model that one wants to keep around, even after having developed a detailed model for explanatory, communicative purposes.

The sequential encoding scenario constrains the ontological freedom of the concept map. It is easier to imagine gradually elaborating from concept map to the model if it were true that the concept map ontology was a strict abstraction of the model ontology. However it is not necessary. That is, the mapping of interactions expressed at the concept map to those in the model may be one-to-one, many-to-many, one-to-many, or many-to-one. A concept map is not just a sparser representation of a model. Sometimes the concept map might contain additional information about the system that never goes into the final model, as the concept map ontology allows one to represent more than what one might be able to say in the detailed modeling ontology. The argument against sequential encoding is that of ontological incompatibility.

The *sketch mode* in the current version of Garp supports the sequential encoding scenario. It is plausible that the modeler might go back and forth between the sketch mode and the qualitative modeling mode; however, the environment does not provide direct support for connecting the sketch and the qualitative model.

In the parallel encoding scenario, the software must provide facilities for keeping concept map and model in sync as they evolve. To implement such tight coupling between the concept map and the model, we need an analysis of relationships between them, which amounts to answering the questions: 1) What are the ontological relationships between elements (nodes and edges) in the concept map and the model? and 2) What kind of activities relate the elements in the concept map and the model? The answers to these questions provide the software with the knowledge required to connect the models. As a first start, the modeler can manually annotate such connections. It is an empirical question for future research to see what aspects of these can be automated and benefit the modeler by automatically pointing out incompleteness and mismatches.

5 Relating Concept Maps to Models

Concept maps draw their power from their lack of representational constraints. This freedom lets one create inconsistent diagrams and mix together causal, structural and other types of information with minimal formal syntax. In addition, one can include components that communicate the scope of a model even though those details will exist only implicitly in the formalized version. In this section,

[4] http://hcs.science.uva.nl/projects/NNR/
[5] http://cmex.ihmc.us/

we examine the relationships expressed in a concept map, how these relationships translate into an encoded model, and the utility of maintaining explicit links between the two representations.

Concept maps can take many forms and encode several types of knowledge: UML diagrams, organizational charts, flowcharts, and so on. To focus the discussion, we emphasize concept maps built as outlines for a causal model (qualitative or quantitative). We ground our discussion in the concept maps built in the CMEX project and those built by Bredeweg's group in the NatureNet Redime project. Although these maps cover a broad scope of topics, ranging from autonomous spacecraft control to river Mesta's ecosystem, we posit that they contain six distinct classes of knowledge: *causal, spatial, mereological, taxonomic, control* and *parametric*. Each type of knowledge manifests either as nodes or as edges in a concept map. We also discuss where the knowledge ends in a qualitative model built using the QPT ontology in the discussion below:

1. *Causal:* Causal knowledge is a key part of explanations, and manifests in relationships such as "causes", "effects", "increases", and "is related to". These relationships map onto qualitative proportionalities and influences, but one can also specify more complex causal relationships like "consumes", "produces", and "regulates," that map onto processes. Relationships such as "enables" and "prevents" capture causal knowledge that becomes preconditions and quantity conditions in a qualitative model. In addition to those specified, we also include temporal relationships like "before", "after", and "during" in this causal category as they often related to a vague causal knowledge.
2. *Spatial:* This type of knowledge captures the spatial layout of entities in the modeled system. Explicit spatial relationships include "above", "below", "inside", "aligned", and so on. While encoding a qualitative process model, one may translate these relationships into preconditions for model fragments as they place limits on which entities can interact with one another.
3. *Mereological:* This type of knowledge describes the part-whole relationships between entities in the system and is expressed by relationships such as "consists of", "contains", and "includes".
4. *Taxonomic:* Taxonomies describe the type information for objects, which manifests as a subtype hierarchy in Garp. Defining specific objects as instance of general types enables the reuse of model fragments. One may describe these relationships with terms like "is a", "type of", "member of", "example of" (for class–instance hierarchies), and so on.
5. *Control:* These relationships introduce control flow into the concept map. For example, one can include a node that determines which of two outcomes will happen. Often control knowledge gives an explicit statement of preconditions and quantity conditions.
6. *Parametric:* Parametric nodes and edges let one introduce modeling abstractions like parameters of interest at the concept map level itself. Ideally, these objects appear directly in the encoded model. In concept maps, such relationships may exist as nodes that represent numeric quantities or edges that represent measurement operations.

As mentioned at the beginning of this section, we are restricting our goal to knowledge contained in causal models. For instance in domains like design, teleology, economics and aesthetics might be some of the other types of knowledge that are relevant to model building. To highlight these relationships, we appeal to the specific examples shown in Figure 3. This concept map describes the river Mesta's ecosystem [Uzunov et al. 2006] and contains seventeen distinct edge labels. We place these labels into the above categories as follows.

1. *Causal:* produces, provides, stimulates, regulates, consumes, feeds on, regulates, influences
2. *Spatial:* inhabit, provides habitat for, lives on
3. *Mereological:* consists of, has, contains
4. *Control:* determines the type of
5. *Parametric:* is measured by

The relation "is profited by" fails to fit in any of the delineated categories. However, consider the statement "particulate organic matter is profited by bacteria." This claim is somewhat misleading as the bacteria consume the particulate organic matter, which defines a process relationship similar to "feeds on" between these two entities. We used the concept map from the river Mesta study to show that many relationships specified in a concept map fall within a limited set of categories. The taxonomic relations do not show up in the concept map as they are modeled separately in Garp. In the next section, we examine the encoding operations associated with these types of knowledge.

6 Operations between Concept Maps and Models

In this section, we describe the operations that a modeling environment needs to have to support the parallel encoding model. We have not built this environment yet. After creating an initial concept map, one can begin the iterative process of model and concept map revision. At this point, the concept map itself becomes a key part of the user interface. Selecting a node will reveal an entity-specific dialog with which one can define either a type or an entity. In the former case, one specifies the properties of the type, which assumes the name of the node. In the latter, one either selects a type for the entity, or, both defines a new entity type and labels the node as an instance of the type. If a taxonomic edge connects two nodes, one can infer the type and properties of the child. In addition, if the concept map lacks an edge between a distinct entity type and its instantiation, the modeling environment can add it automatically. This action synchronizes the concept map and the encoded model and is an important tool for revealing relationships that were initially implicit but that

became explicit during the formalization process. More plainly, this activity helps one see their previously implicit knowledge, which may lead to a better understanding of the system and better modeling habits in the future.

One can also select the edges of the concept map and assign them to any of the six specified categories. Structural and spatial edges establish contexts and may imply a need for a transport process. For instance, a detailed cell model may have a pool of RNA within the nucleus and a pool of RNA in the cytoplasm. A model that incorporated mechanisms for transcription and translation would require a process that shuttles the RNA through the nuclear membrane. If the concept map contains mereological and boundary information, then the modeling environment could readily infer the necessity of such a process and remind the modeler to include it. If one labels an edge as taxonomic then the environment can relay properties from the source node to the source (an entity type) to the target (either more specific type or an instantiated entity). Finally, labeling an edge as causal will call up the process editing dialog. First, the environment ensures that the source and target of the link are both entities. If not, the interface will prompt the modeler to add an instantiation of the appropriate type. Afterwards, both entities will appear in the process definition, which allows the inclusion of other entities as participants. In the simple case, the modeler defines the process between two entities and moves to another relationship. If, however, other entities are introduced, the program must bring the concept map up to date by adding process links between the newly related nodes.

Next we present a catalog of operations between concept maps and models. The software must be aware of these, and can help the modeler annotate and connect the concept map and the model. This is a coarse representation of the types of activity links that exist between concept maps and models.

1. *Typing:* The modeler takes a node or an edge in the concept map and provides the type information for it from the ontology(e.g., identify something as a process, quantity, or an influence). At this point, the software can use templates associated with the types to point out the information that is needed to fully describe it in the modeling ontology. Further, local constraint satisfaction could propagate this information and anticipate the types of other nodes and edges connected to the object.
2. *Elaboration:* The modeler takes a node or edge in the concept map and decides to explode it and model it in further detail. The software makes sure that the internal and external connectivity to this object is maintained. Other than this, one can freely elaborate the object in any way allowed by the modeling ontology. This procedure is similar to the *model containers* idea in ModKit [Bogush, Lohmann and Marquardt, 2001].
3. *Filtering:* This operation has the modeler specify the elements in the concept map that will not be described in the simulation model. This could be because the detailed modeling ontology cannot encode those elements, or they might not be relevant to the task at hand.
4. *Annotation:* We allow this as a catchall relationship between the concept map and the model, where the

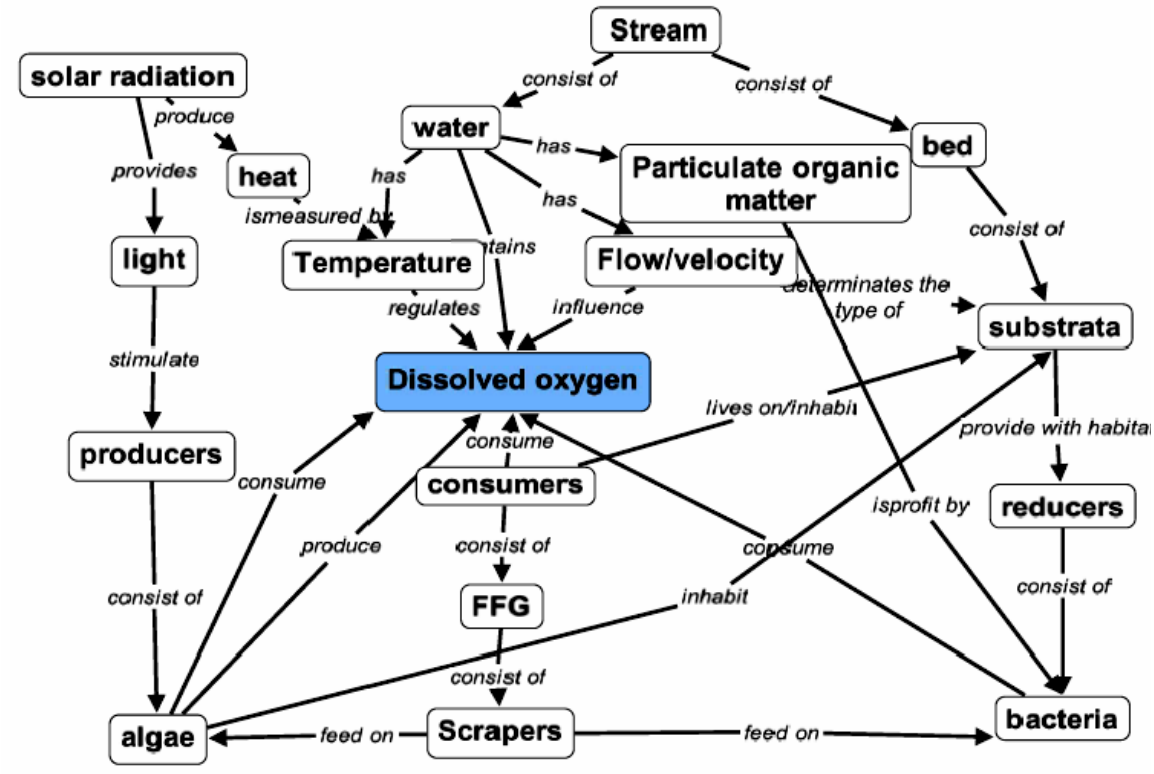

Figure 3. Concept map of the River Mesta ecosystem [from Uznov et al, 2006]

modeler can select a subset of the concept map and connect it to the model without specifying the detailed relationship between the elements.

The above list makes it possible for the modeler to explicitly connect the concept map to the model. Six types of knowledge in concept maps and four types of operations going from concept maps to models, gives a set of twenty-four connection types. Further modeling constraints might make it possible for the environment to automatically detect mismatches and/or incompleteness in the concept map or the model. Reasoning from the model fragments and assumptions [Falkenhainer and Forbus, 1991; Nayak, 1992] might play a key part in operationalizing these constraints.

7 Conclusions

Building models is hard. We argue that a tight integration of the conceptual and detailed modeling processes in the modeling environment can facilitate modeling. We claim that there are six classes of knowledge that are described in concept maps: causal, spatial, mereological, taxonomic, control and parametric. We describe four types of operations that connect concept maps to models: typing, elaboration, filtering and annotation. We believe that this raises interesting research questions about how to provide automatic support for these operations in the modeling environment. Implementing these ideas in a modeling environment like Garp or Stella will provide insights about their usefulness, and we hope that this paper sparks a conversation about building better modeling environments.

Acknowledgments

This work is supported by Artificial Intelligence Program of the Computer Science Division of the Office of Naval Research and by Grant No. IIS-0326059 from the National Science Foundation. Praveen Paritosh would like to thank Ken Forbus, Tom Hinrichs, Matt Klenk, Bert Bredweg, and Andrew Lovett for insightful discussions. Will Bridwell would like to thank Pat Langley, Dorrit Billman, Stuart Borrett, Bert Bredeweg, Anders Bower, and Desiree Tullos for conversations about conceptual modeling.

References

Bogusch, R., Lohmann, B., Marquardt, W. (2001). Computer-aided process modeling with ModKit, *Computers and Chemical Engineering*. Volume 25, Number 7, pp. 963-995.

Bredeweg, B., Salles, P., Bouwer, A., Liem, J., Nuttle T., Cioaca, E., Nakova, E., Noble, R., Caldas, A., Uzunov, Y., Varadinova, Y., Zitek, A. in press, Towards a Structured Approach to Building Qualitative Reasoning Models and Simulations, *Ecological Informatics*.

Bridewell, W., Sanchez, J., Langley, P., Billman, D. (2006). An interactive environment for the modeling and discovery of scientific knowledge. *International Journal of Human–Computer Studies*, 64, 1099–1114.

Buzan, T. (1991). The Mind Map Book . New York: Penguin.

Falkenhainer, B. and Forbus, K. (1991). Compositional Modeling: Finding the Right Model for the Job. *Artificial Intelligence*, 51, 95-143.

Farabee, M., (2001). On-line biology textbook, retrieved from http://www.emc.maricopa.edu/faculty/farabee/

Forbus, K., Carney, K., Sherin, B., Ureel L., (2004). VModel: A Visual Qualitative Modeling Environment for Middle-school Students, In *Proceedings of the 16th Conference on Innovative Applications of Artificial Intelligence.*

Forbus, K., Lockwood, K., Klenk, M., Tomai, E., and Usher, J. (2004). Open-domain sketch understanding: The nuSketch approach. To appear in *AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural*, October, Washington, DC.

Forbus, K. and Usher, J. (2002). Sketching for knowledge capture: A progress report. In *Proceedings of IUI 2002*, San Francisco, California.

Forrester, J. W. (1961). Industrial dynamics. Waltham, MA

Foss, B. A., Lohmann, B., Marquardt, W. (1998). A Field Study of the Industrial Modeling. *J. of Process Control*, 8(6):325--338.

Haefner, J.W. (2005). Modeling Biological Systems: Principles and Applications, Springer.

Jackson, M. (1995). Software requirements & specifications: a lexicon of practice, principles and prejudices, ACM Press/Addison-Wesley Publishing Co. New York.

Jorgensen, S. (2001). Fundamentals of Ecological Modelling, 3rd edition, Elsevier.

Leake, D., Maguitman, A., Reicherzer, T., Cañas, A., Carvalho, M., Arguedas, M., Brenes, S., Eskridge, T., (2003). Aiding Knowledge Capture by Searching for Extensions of Knowledge Models, *Proceedings of K-CAP ´03*, October 2003, Sanibel Island, Florida.

Leelawong, K. (2005). Using the Learning-by-Teaching Paradigm to Design Intelligent Learning Environments, Doctoral Dissertation, Vanderbilt University.

Levins, R. (1966). The Strategy of Model Building in Population Biology, *American Scientist* 54: 421–431.

Lohner, S., van Joolingen, W., Savelsbergh, E. (2003). The effect of external representation on constructing computer models of complex phenomena, *Instructional Science* 31: 395–418.

Moran T.P., Carroll J.M., (1996). Design Rationale: concepts, techniques, and use.

Novak, J. D. & A. J. Cañas. (2006). The Theory Underlying Concept Maps and How to Construct Them, Technical Report IHMC CmapTools 2006-01, Florida Institute for Human and Machine Cognition.

Nayak, P. P., (1992). Automated Modeling of Physical Systems, Doctoral Dissertation, Stanford University.

Rittel, H., and M. Webber; "Dilemmas in a General Theory of Planning" pp 155-169, *Policy Sciences*, Vol. 4,

Elsevier Scientific Publishing Company, Inc., Amsterdam, 1973.

Sins, P., Savelsbergh, E., van Joolingen, W. (2005). The Difficult Process of Scientific Modelling: An analysis of novices' reasoning during computer-based modeling. *International Journal of Science Education*, Volume 27, Number 14, Number 14/18 November 2005, pp. 1695-1721(27)

Tomai, E., Forbus, K., and Usher, J. (2004). Qualitative spatial reasoning for geometric analogies. *Proceedings of the 18th International Qualitative Reasoning Workshop.*

Uzunov, Y., Elena Nakova, E., and Varadinova, E. 2006. Textual description of river Mesta case study, Naturnet-Redime, STREP project co-funded by the European Commission within the Sixth Framework Programme (2002-2006), Project no. 004074, Project Deliverable Report D6.3.1.