Helping children become qualitative modelers

Kenneth D. Forbus
Qualitative Reasoning Group
Northwestern University
1890 Maple Avenue
Evanston, IL, 60201, USA
forbus@northwestern.edu

Abstract

Early science education is essentially qualitative. Children must learn causal theories: what kinds of things happen, when they happen, and what their consequences are. Although modeling is a key skill in scientific reasoning, traditional modeling techniques do not express causal theories explicitly, and rely on higher mathematics (such as algebra) that children do not learn until later. We believe that qualitative modeling techniques developed in AI can become an important tool for early science education. This paper describes a notational system we are building that enables middle-school students to create and use qualitative models. We use a restricted concept map language based on qualitative process theory to provide a student-friendly visual modeling language. Interactive qualitative reasoning techniques and analogical coaching will be used to help students understand and refine their models. The basic design of the system will be outlined, as well as experiences with its use with students in the Chicago Public School system.

1 Introduction

Early science education is essentially qualitative. Children must learn causal theories: what kinds of things happen, when they happen, and what their consequences are. This early learning provides a solid conceptual foundation for later science education, as well as being directly useful in helping them learn to deal with their world. Computers have had less impact in early science learning than in more advanced instruction. One reason is that many educational software efforts in science have relied on computing's traditional strengths in numerical and mathematical modeling. However, mathematics, which starts becoming important in high school science and often takes center-stage in college science instruction, is not available to young children. A different approach is needed to create software that is more suitable for the needs of children learning science.

This paper argues that ideas developed in the qualitative reasoning community provide a solid conceptual framework for creating software for early science education. Our goal is to create a modeling system that is so flexible and so easy to use that it will become to modelers what word processors are to writers and spreadsheets are to accountants. Section 2 argues that students should learn qualitative modeling as part of learning scientific modeling. Section 3 describes a student-friendly notional system we have created for expressing models in qualitative process theory [Forbus 84], and a software system we have created, called *VModel*, which helps students use this notation. Section 4 outlines work in progress on using qualitative reasoning and analogical processing in coaching in VModel, and Section 5 describes our work with these ideas in the Chicago Public Schools. Section 6 discusses related projects, and Section 7 closes with plans for future work.

2 Why qualitative modeling for students?

Why do we want students to become modelers? Modeling is a central skill in scientific reasoning. Helping students to articulate models of a domain, and to refine them through experience, reflection, and discussion with peers and teachers, can lead to deeper, systematic understanding of science [Collins 96].

Why do we want students to become qualitative modelers? Scientific modeling is often taught in ways that focus on mathematical models, and most modeling software uses numerical analysis to derive its results. Younger students, in elementary and middle school, typically do not yet have the background necessary to learn from such activities. But why must scientific modeling be taught only this way? We believe that solid instruction in qualitative, conceptual knowledge is equally important. Qualitative modeling focuses on causal, conceptual knowledge – exactly the kind of knowledge that younger students are trying to master. Qualitative reasoning formalisms provide the expressive power needed to capture the intuitive, causal notions of many human mental models [Forbus 97]. Qualitative representation techniques also can express kinds of modeling knowledge that traditional formalisms do not, such as the conditions under which a model is appropriate. By using such formalisms, software can be created that supports a broader range of modeling activities.

These properties suggest that qualitative modeling can provide a solid foundation for scientific modeling, especially for younger students. (Even for older students, results in physics education [Camp 94] suggest that a solid conceptual foundation is a crucial element for success in subsequent traditional training.) How can we help students become qualitative modelers? Let us start to answer this question by examining how models are used and created.

Models provide a means to externalize thought. External representations help reduce working memory load, allowing students to work through more complex problems than they could otherwise. External representations also help them present their ideas to others for discussion and collaboration. This implies that the language in which models are expressed must be kept as simple as possible. The conceptual difficulties students face are complex enough; they do not need the additional complexity of (to them) arcane syntax and obscure technical terminology. While the representational ideas of qualitative modeling are ideal

To appear in: Journal of the Japanese Society for Artificial Intelligence, special issue on Intelligent Media in Educational Systems, 2002.

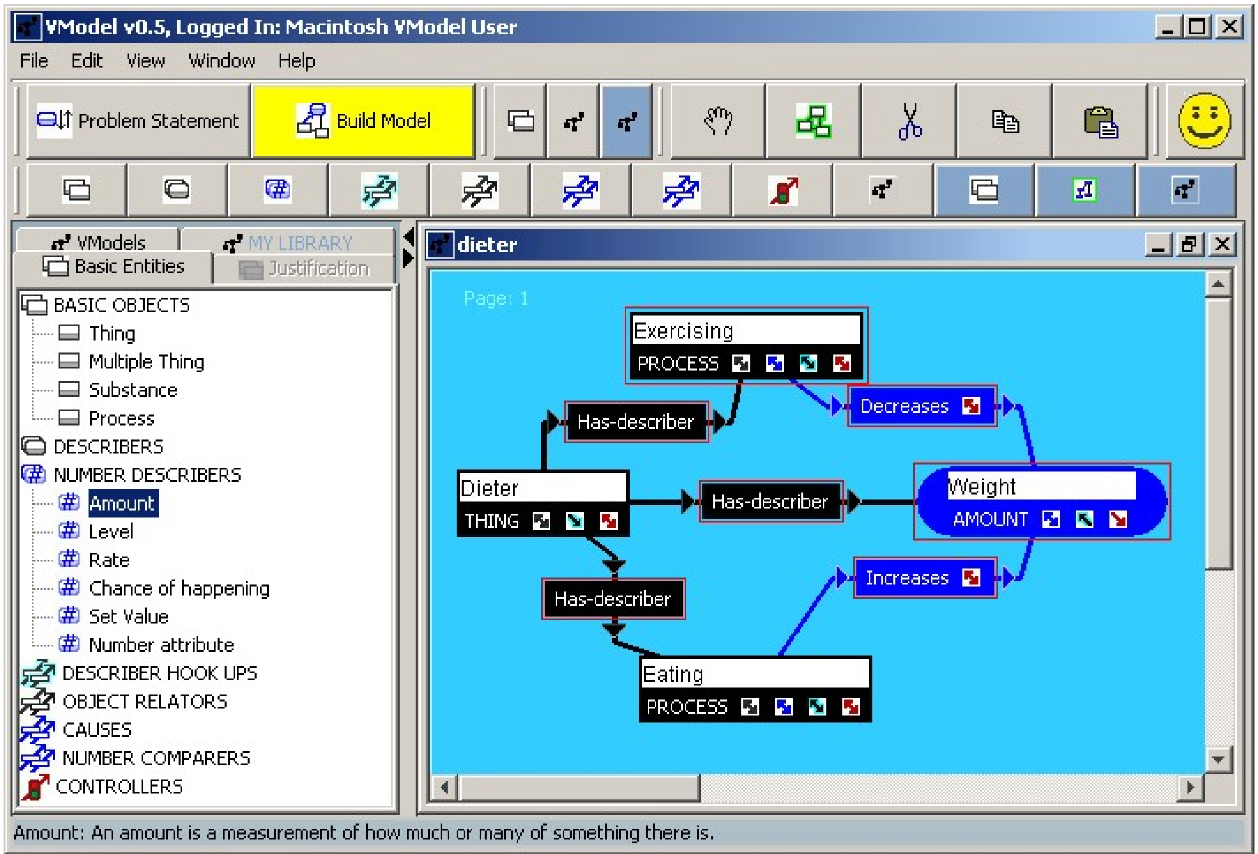


Figure 1: The Vmodel interface

for science education, the predicate calculus formalisms typically used to express them are not.

Our solution to this dilemma has been to develop a *visual representation language*, based on concept maps [Novak 84], for qualitative modeling. This language provides a student-friendly way to express qualitative models, for their own thinking and to facilitate communication with others. The underlying semantics for this language is based on qualitative process theory, which facilitates creating software that supports students in modeling. We turn now to describing this visual language and the VModel software which supports it.

3 Visual qualitative modeling and VModel

We started by carefully examining previous visual languages for modeling in science and science education. These can be grouped into three families: (1) *Concept Map notations*. Concept maps describe structural and functional properties and relationships between entities and ideas. Typically, concept maps express global or time-invariant information. (2) *Dynamical Systems notations*. Forrester's version of system dynamics [Forrester 96], Bond graphs, and software systems such as STELLA and Model-It [Jackson 96] provide graphical languages for expressing differential equations for continuous systems. (3) *Argumentation environments*. Belvedere [Suthers 97] and the Collaboratory Notebook [Edelson 96] are examples of computer systems which use graphical conventions to help students gather, create and reason about evidence and arguments for and against hypotheses.

Each of these notations enables students to express some aspects of modeling, but none of them alone is sufficient. For example, concept maps can in theory be used to express anything, but their lack of a standardized semantics makes it difficult for students to understand each other's concept maps and extremely difficult to create software that detects whether or not arguments and models are well-formed [Canas 95]. Dynamical system notations do not express the conditions under which a given model is applicable. Argumentation environments treat as atomic what would be whole complex structures in the other notations, which limits their ability to scaffold students.

Even taken together, these notations neglect three key issues in understanding the art of modeling: (1) *The importance of broadly-applicable principles and processes*. Existing educational modeling systems treat each modeling task as a new problem, with no connection to other situations. This misses the opportunity to help students see that the same principles and processes operate across a broad range of situations. For example, the basic idea of heat flow is relevant to chemistry, biology, atmospheric physics and many other areas that, on the surface, appear unrelated. Existing modeling systems do not help students see the importance of creating a systematic body of knowledge, as opposed to a series of ad hoc explanations concerning specific systems. (2) *Understanding when a model is relevant*. A crucial skill is knowing when a model is appropriate.

For example, treating plant life as essentially infinite is fine in many predator/prey models but inappropriate when modeling an island or space station. Existing educational modeling systems do not address this issue and thus do not help students connect their models to real-world concerns. For example, public policy debates often rest on the correctness of assumptions underlying competing models (e.g., is global warming really occurring? How much refuge land is needed to preserve biodiversity?). (3) *Qualitative understanding of behavior*. Modeling systems tend to be numerical (e.g., STELLA), although sometimes including a qualitative layer on top to simplify model creation (e.g., Model-It). Understanding how numerical data plots depict behavior is certainly an important skill. However, using these tools requires that students think in terms of detailed mathematical ideas in addition to the conceptual level, and they must provide significant amounts of numerical data. These requirements can be serious distractions for students who have not yet mastered a phenomenon conceptually.

The theories, representations, and reasoning techniques developed in qualitative reasoning research provide most of the pieces needed to address these problems. Enabling and encouraging students to create their own domain theories should help them understand the broad applicability of scientific principles and processes. The techniques of compositional modeling [Falkenhainer 91] provide the expressive power needed to state modeling assumptions and reason about relevance. Qualitative modeling provides formalisms for expressing intuitive, causal models and the reasoning techniques needed to generate predictions and explanations from them for helping students see the consequences of their ideas. Making these formalisms available through a visual notation is, we believe, the missing piece that will make this power accessible to young students.

3.1 The design and architecture of VModel

The VModel qualitative modeling environment combines good ideas from all three of the notations described above.
Its organization is based on concept maps, but with some very strong restrictions. As usual, nodes represent entities and properties of entities. However, each node has a specified type, such as `Thing` or `Process`. These types are drawn from a general ontology provided with the system. This ontology can be extended by students. As usual, links represent relationships. However, the labels that can be used on links are drawn from a fixed set of relationships.

These restrictions provide a clearer semantics than traditional concept maps have. In traditional concept maps, any path is intended to be a proposition, i.e., a natural-language statement that is true about what is being described. With these restrictions, links in our concept maps can be identified with propositional logic statements involving binary relations. This makes software coaching more feasible than in traditional concept maps. It also enables students to link their own propositions together, to record the reasons for their beliefs. Figure 1 shows what the interface looks like.

These constraints address the tradeoff between providing freedom of expression versus scaffolding for students. Providing their own names for entities and properties enables them to express their ideas more accurately (e.g., "temperature" versus "hotness" versus "cold"). Requiring students to select a general type for entities and properties helps coaching software figure out which is which, and avoids the need to do natural language understanding on their typed phrases. Enabling students to extend the ontology will, we hope, provide them with additional incentive to generalize the concepts in their models. Restricting links to a fixed set of relationships provides a powerful scaffold for students, ensuring that their ideas are at least in the ballpark in terms of form of argument[1]. It also forces students to enter a community of modelers, enabling their ideas to be compared and contrasted with those of others more easily.

The other basic feature of our design is the use of a *Model Library*. All of the models a student (or a group of students) have built are organized in their Model Library. The Model Library also contains the abstractions they create based on their modeling of specific situations, i.e., their domain theory. The contents of their Model Library thus represents their evolving understanding. In addition to being a portfolio and support for reflection, the Model Library is being designed to facilitate the construction and reuse of descriptions, in order to help students construct general principles and laws.

We want VModel to be useful in all stages of qualitative modeling, from gathering and summarizing the phenomenon to be explained to initial model formulation to refinement via testing the consequences of the model against data. Given this large range of tasks, factoring the representation functionally becomes very important. We divide our notational system into three interconnected parts:

- *Situation maps* express the structural properties of a situation, linking the description of the system or phenomena to objects, relationships, and processes in the student's domain theory.
- *Causal maps* express the causal relationships between continuous parameters, using the vocabulary of qualitative process theory.
- *Evidence maps* express the rationale for the choices in the situation maps and causal maps in terms of links to other knowledge and information sources, and annotations that express their thinking in a free-form way.

In the rest of this section, we describe each of these maps in turn, then discuss the Model Library design, modeling workflow, and coaching.

3.2 Situation maps

In the situation map, students describe the entities in the modeled world, as well as structural and configural relationships among them. These entities and relationships give rise to physical processes which, in a well-developed model, provide the causal map.

The ontology used in the situation map is based on familiar AI representation practice, although some of the names have been changed to be student-friendly, based on our classroom experiences. All entities in the situation map are either basic entities, which can be introduced on their own, properties of entities, or an observed change. Entities and properties are related by the HAS-DESCRIBER relationship, whereas entities and the processes they participate in are indicated by DOES. Quantities are represented by NUMBER-DESCRIBERs, with ordinal information expressed via comparatives. Students are

[1] For example "Pixies do it" and other anthropomorphic arguments are simply not expressible.

allowed, and indeed encouraged, to extend the ontology of entity and attribute types. We provide a small vocabulary of relationships to express configural information (e.g., TOUCHES, CONTAINS, PART-OF), and a simple vocabulary for specifying conditions under which processes can occur (aka CONTROLLERS). Teachers and curriculum developers are allowed to extend the configural relationships vocabulary as necessary. Figure 2 illustrates a relational statement expressed using these conventions. (Highlighting is automatically used to visually tie properties to entities that they belong to, something we found important in classroom experimentation.)

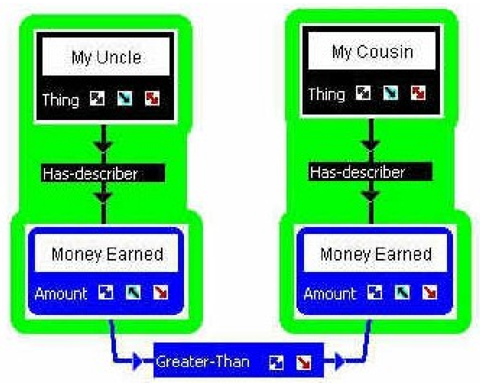


Figure 2: a simple relational statement in VModel.

3.3 Causal maps

The vocabulary for causal maps is drawn from qualitative process theory. Specifically, we use QP theory's qualitative mathematics, the language of *influences*. Influences are particularly appropriate because of their compositional nature: Each influence can be stated as a link, and the set of such links provides the set of influences on a parameter in a student's model. Here are the "student-friendly" relationships we are using for expressing influences:

- `Affects`: Indicates the causal direction between two parameters without any commitment to its nature.
- `Increases/Decreases`: Indicates an integral connection between two parameters, i.e., heat flow decreases the heat of its source and increases the heat of its destination (direct influences, I+/I-, in QP theory).
- `Linked-to/Linked-opposite-to`: Indicates functional dependence between two parameters, i.e., the heat of something determines its temperature. These are qualitative proportionalities, in QP theory, with `Linked-to` expressing $\propto_{Q+}$ and `Linked-opposite-to` expressing $\propto_{Q-}$.
- `Linked-with/Linked-opposite-with`: Indicates that the two parameters are changing at the same time, in the same/opposite direction, but the direction of causality if any is unknown.

We believe that the ability to express partial relationships and make causal arguments about relations between entities in the world makes QP theory an ideal modeling formalism for middle school and high school students. Unlike the relations vocabularies in the situation map, which can be varied by teachers and curriculum designers, the vocabulary for the causal map is unchangeable, to support qualitative reasoning with the student's model.

3.4 Evidence Maps

Evidence maps record the reasons for the choices made in constructing a model. Evidence maps justify behavior in terms of aspects of the model, and aspects of the model in terms of experiments, hypotheses, and other sources of data. This is an especially important feature to include in software for young students, as they often do not understand that scientific ideas are often theories or explanatory constructs and as such need to be supported. Evidence maps are implemented as another restricted concept mapping language, where observations and data serve as nodes that can be linked to nodes in the rest of the concept map.

3.5 The Model Library

The true power of modeling arises when students can use concepts they developed in earlier modeling exercises to tackle more complex modeling problems. Few modeling tools explicitly support this kind of reuse and abstraction. Yet we believe that this facility will provide valuable encouragement for the systematization of a student's knowledge. For example, maintaining a library of models and abstractions derived from them should facilitate being able to transfer ideas from one problem to another, or even one domain to another.

The Model Library contains two kinds of information. First, it includes a *portfolio* of all of the modeling projects the student (or a group of students) have tackled. The ability to refer back to previous exercises promotes reflection, and, as discussed later, we plan to use analogical processing techniques to nudge students to think in terms of previous problems when appropriate. The second kind of information in the model library is the *catalog* of entities, properties, and relationships that can be used to build models. This aspect of the library uses a construction kit metaphor, where students use the "building blocks" of these elements to build new models. Unlike traditional construction kits, however, students are able to create their own building blocks, extending the Model Library with new entity and property types that reflect their own growth in understanding. It might at first seem that these two aspects of the library should be separated. We combine them into a single unit in the system because we have student models automatically cross-indexed under the catalog elements that they use, promoting further reflection and transfer.

The underlying formal ontology for objects and processes is that of QP theory. The Model Library's catalog represents the student's domain theory, with the elements in it being model fragments.

As a student progresses, their Model Library will contain (aside from a relatively small set of primitive relationships and object types built in) student-constructed, student-friendly descriptions of model fragments, arranged in catalogs of objects, parameters, and processes. Items in these catalogs are constructed using the same visual notations found in causal, situation, and evidence maps. The only difference is that some of the objects in these model fragments are *connectors* which must be hooked up in order to use that item. Fragments of a causal map are also associated with these new building blocks, so that when they are hooked up, those relationships are added to the causal map. New building blocks are created by selecting a subset of an existing model and specifying what aspects should be turned into connectors, and what information should be kept in the fragment.

The catalogs in the Model Library are organized into trees for types of objects, relationships, and processes. Each tree is organized via inclusion, e.g., a population of elk is a kind of population, which is a kind of multiple-thing, and convection is a kind of heat flow, which is a kind of process[1]. The tree model facilitates generalization. By explicitly supporting more intermediate states of knowledge, we hope to encourage students to use the software for brainstorming.

3.6 Supporting the modeling process

Modeling involves several different activities. We communicate and support this by organizing the software among several modes, each of which corresponds to a modeling activity. These are *stating the problem*, *modeling*, and *justification*. We describe each in turn.

Problem statement. Students are always asked to state a problem that their model is trying to solve. That is, given what they are trying to model, they are asked to identify one continuous parameter of the system, and indicate how they believe that it will change. The purpose of the model becomes providing an explanation for this belief. We discovered through preliminary experiments that such structuring is very important for students. Otherwise, they tend to treat modeling as a "brain dump", putting in every entity and relationship that they can think of, whether or not they are mutually consistent. Having a guiding question provides a focus, a means of ascertaining when they have a successful model.

Modeling. This mode is where students create their situation and causal maps, using the representations outlined above. Most of the student's time is spent here.

Justification. This mode is where students build an evidence map that justifies the contents of their situation and causal maps. By incorporating this process as a distinct mode, we underscore its importance for students and encourage reflection about their model as a whole.

3.7 Coaching

In an ideal world, powerful AI software using the latest qualitative and analogical reasoning technology would be built directly into the modeling environment. Ours is not an ideal world. Our collaborating schools are part of the Chicago Public School system, and have few computational resources. This forces us to keep the software small, simple, and cross-platform, since our collaborators have a mixture of PCs and Macs.

The classroom software is written in Java, and provides the facilities students need for creating and editing models and domain theories. (Printing facilities become especially important when computers are scarce, and we are adding support for generating web pages of student models automatically as well.) We have built in simple coaching facilities, the modeling equivalents of spelling correctors and grammar checkers. For example, when a student links a causal connection directly to a process instead of its rate, the software suggests using the rate instead.

4 Classroom experiences

The classroom portion of this research is being conducted as part of the NSF Center for Learning Technologies in Urban Schools, (LeTUS) a partnership involving Northwestern University, University of Michigan, and the Chicago and Detroit Public School systems. The Center is developing inquiry-based middle school science curricula. These curricula are being developed in *work circles*, which involve researchers, teachers, and school administrators in developing and adapting materials for classroom settings.

We helped CPS teachers develop two curricula. One curriculum concerns heat and temperature, in which students consider alternate energy resources for homes (cf. [Linn]). In collaboration with Marcia Linn's group at Berkeley, we are adding complementary simulation-based activities to their successful thermal curriculum [Linn 91]. These activities use self-explanatory simulators [Forbus 90] to enable students to explore the consequences of their design choices in making a solar house. The other curriculum concerns ecosystems, using as motivation the creation of a life support system for a Mars colony. This provides an arena for students to explore the requirements of life and how ecosystems work, using simulation experiments.

Our design process for the visual notation was guided by pencil and paper studies carried out in CPS classrooms. The first pilot studies with the software took place in Winter and Spring of 2001, in a bilingual magnet school in Chicago. Eleven sixth graders (aged eleven and twelve) worked together, in groups of two or three, creating models related to the Mars curriculum. A second study is underway (Fall 2001/Winter 2002), also using the Mars curriculum. In a seventh-grade classroom, students are doing a combination of class-wide and small-group modeling exercises. Students working in small groups are asked to discuss and justify their models in front of the entire class. Data is being gathered in the form of videotaping, clinical interviews, and portfolios of models constructed.

While data is still being collected, there are some encouraging observations that can already be made:

- *Naturalness of the visual language*. After some tuning (e.g., using words like "describer" instead of "attribute"), student questions tend to focus on contents of the models, rather than on how to use the primitives of the modeling language.
- *Generalization in modeling*. Even though there have not been many opportunities yet in the curriculum to make generalizations based on their models, we are finding that students do indeed start exploiting abstractions. One student for instance, used their model of gazelles and grass to model the interaction between lions and gazelles via the appropriate substitutions. This is not an easy thing for students to grasp. In one classroom discussion, students were working out what to name a model they had created that described an astronaut's weight gain or loss in terms of their caloric intake and their exercise and metabolic needs. Could the model be used to explain more than just astronauts? How could it be made truly general, if it wasn't already? One girl ventured "I think it should be called 'the calorie cycle' because you could

[1] Actually, in logical terms the organization is a lattice, since there is multiple inheritance: Population may inherit from both multiple-thing and a student-introduced type called living-thing. However, we think that having the same entities appear in multiple locations is a simpler visual interface model.

take out ASTRONAUT and replace it with DOG, and that model would explain both." That is the kind of insight that we want them all to attain.

5 Plans for additional coaching

As noted above, currently VModel only does very simple coaching, catching what are the modeling equivalent of grammatical errors. We plan to significantly expand the coaching facilities, in three ways. First, we are designing a simple, within-state qualitative simulation capability that can generate explanations about changes implied by the model, producing a natural language hypertext explaining how the behavior was derived. We expect that seeing these arguments will solve a problem we saw in our classroom experiments, when students would become confused as to the difference between the types of causal links. It will also provide an automatic test as to whether their model correctly predicts the phenomena in the problem statement, which will help provide a sense of closure to their work.

The second coaching facility we are designing is analogy-based [Forbus 95], to provide feedback based on normative models. (Normative models are authored using the same software, so that teachers and curriculum developers can add content without us being involved.) The candidate inferences of a comparison of a student model to a normative model can provide suggestions for what a student might want to think about in order to improve their model [Collins 82].

The third coaching facility we are designing is for assessment and class-level feedback. It is based on a *distributed coaching* model [Forbus 99], where lightweight coaching happens "on-board", while students and teachers access more powerful coaching facilities via email facilities built into the software. The idea is to gather student models within a classroom and compare them with each other. By using SEQL [Kuehne 00] to group them into categories, we hope to uncover common patterns of misconceptions within a classroom. By knowing which students belong to which clusters, we may also be able to suggest interesting discussion groups, based on shared or differing models. Teachers we have discussed this with find the potential quite interesting; It is something that is difficult for them to find time to do, given their workload.

6 Related Work

The use of visual modeling environments has a long history in science and technology, although most have been aimed at professional scientists and engineers. The impressive design environment created at University of Tokyo [Umeda 96], for example, shows how qualitative modeling can play an important role in engineering.

We know of two projects that are very similar to ours in both spirit and approach. One is the UWF Quorum project [Canas 95, Canas 98], which used concept maps to let student express and share a wide variety of ideas, both within their schools and with students in other countries. Quorum's success encouraged us to consider the use of restricted concept maps as a visual notation for qualitative modeling. Their use of an "artificial idiot", the Giant, is an approach to coaching we are considering, to exploit the *persona effect* [Lester 97]. The second is "Betty's Brain" at Vanderbilt [Biswas 01], where they are using qualitative representations in concept maps to foster learning. The task they use, of "teaching" Betty (their software) by building concept maps so that Betty can produce explanations, is inspired. Their qualitative modeling framework uses qualitative mathematics, with tables for composing discrete values to provide qualitative simulation. Their experience with explanations generated via qualitative simulation provides strong evidence that adding such facilities to our system will be useful. Our qualitative modeling framework is richer, incorporating physical processes and a student-extendable ontology of types of entities. We also support the creation of new abstractions from student models, which the Vanderbilt software does not.

7 Discussion

This paper summarizes our work in progress to create a representational system with computer support that we hope will enable middle-school students to learn to be modelers. By using qualitative representations as a formal semantics for a restricted concept map language, we have created a visual notation for modeling. The situation map provides a medium for students to describe their observations, the causal map provides a medium for students to express their hypotheses about mechanisms, and the evidence map provides a medium for students to express the reasoning that underlies their beliefs. These representations, when embedded in a computer system that scaffolds students, provide a powerful language for students to use in creating and refining models. Based on our pilot studies and experiments in progress, we believe that this combination will be very useful for a wide range of students. We are particularly encouraged by the spontaneous uses of the visual qualitative language by students outside of the software that we have observed; it suggests that this way of looking at the world is becoming habitual, not simply tied to using the VModel software.

The pilot studies underway with the current software will help us refine and revise our software. As we add more coaching, we hope that VModel will evolve into a tool that will help students become full-fledged modelers, engaged in the joy of unraveling complex phenomena rather than frustrated by memorizing mountains of isolated facts. By keeping the entry barriers for use as low as possible, we hope to create a tool that will be to modelers what word processors are to writers and spreadsheets are to accountants.

Acknowledgements

This research is supported by the National Science Foundation under the REPP program. The members of the VModel team are Bruce Sherin (co-PI), Leo Ureel (lead programmer), and Karen Carney (Education). Jared Dunne and Rob Harris provided additional programming assistance. We have had the invaluable support of several Chicago Public Schools teachers in developing curricula, and thank them for allowing us to use their classrooms for testing purposes. These teachers are Beverly Miller, Deborah Rogers-Green, Adam Dorr, Carol Scafide, Kiesha Korman, Carlos Rodrigues, and Judy Whitcomb.

8 References

[Biswas 01] Gautam Biswas, Daniel Schwartz, John Bransford & The Teachable Agents Group at Vanderbilt. In press. Technology Support for Complex Problem Solving: From SAD Environments to AI. To appear in Forbus, K. and Feltovich, P. (Eds.) 2001 *Smart Machines in Education: The coming revolution in educational technology*. AAAI Press.

[Camp 94] Camp, C. W. and Clement, J. 1994. *Preconceptions in Mechanics: Lessons dealing with student's conceptual difficulties*. Kendall/Hunt Publishing Company, Dubuque, Iowa.

[Cañas 95] Cañas A. J., Ford K. M., Brennan J., Reichherzer T., Hayes P.1995. Knowledge Construction and Sharing in Quorum. World Conference on Artificial Intelligence in Education, Washington DC, July.

[Collins 96] Collins, A. 1996. Design issues for learning environments. In S. Vosniadou, E.D. Corte, R. Glaser, & H. Mandl (Eds.) *International perspectives on the design of technology-supported learning envrionments* (pp 347-362). Mahwah, NJ: Erlbaum.

[Collins 82] Collins, A., & Stevens, A. L. (1982). Goals and strategies of inquiry teachers. In R. Glaser (Ed.), *Advances in instructional psychology* (Vol. 2, pp. 65-119). Hillsdale, NJ: Erlbaum.

[Edelson 96] Edelson, D. C., Pea, R. D., & Gomez, L. M. (1996, April). The Collaboratory Notebook: Support for Collaborative Inquiry. Communications of the ACM, 39, 32-33.

[Falkenhainer 91] Falkenhainer, B. and Forbus, K. Compositional Modeling: Finding the Right Model for the Job. *Artificial Intelligence*, **51**, (1-3), October, 1991.

[Forbus 97] Forbus, K. and Gentner, D. 1997. Qualitative mental models: Simulations or memories? *Proceedings of the Eleventh International Workshop on Qualitative Reasoning*, Cortona, Italy.

[Forbus 95] Forbus, K., Gentner, D. and Law, K. 1995. MAC/FAC: A model of Similarity-based Retrieval. *Cognitive Science*, 19(2), April-June, pp 141-205.

[Forbus 99] Forbus, K.D., Whalley, P., Everett, J., Ureel, L., Brokowski, M., Baher, J. and Kuehne, S. (1999) CyclePad: An articulate virtual laboratory for engineering thermodynamics. *Artificial Intelligence*. **114**, 297-347.

[Forbus 84] Forbus, K. Qualitative Process theory. *Artificial Intelligence*, **24**, 1984.

[Forbus 90] Forbus, K. and Falkenhainer, B. "Self-explanatory simulations: An integration of qualitative and quantitative knowledge", AAAI-90, August, 1990.

[Forrester 96] Forrester, Jay W. *System Dynamics and K – 12 Teachers.* Lecture: University of Virginia School of Education, May 30, 1996.

[Jackson 96] Jackson, S., Stratford, S.J., Krajcik, J.S. and Soloway, E. (1996). A Learner-Centered Tool for Students Building Models. Communications of the ACM, 39(4), 48 - 50.

[Kuehne, 00] Kuehne, S., Forbus, K., Gentner, D. and Quinn, B.(2000) SEQL: Category learning as progressive abstraction using structure mapping. *Proceedings of CogSci 2000*, August, 2000.

[Lester 97] J. Lester et al., "*The persona effect: Affective impact of animated pedagogical agents*," in Proc. Human Factors Comput. Syst., Mar. 1997, pp. 359--366.

[Linn] Linn, M.C., Houses in the Desert. http://wise.berkeley.edu/WISE/about/houses/

[Linn 91] Linn, M. C. & Songer, N. B. (1991). Teaching thermodynamics to middle school students: What are appropriate cognitive demands? Journal of Research in Science Teaching, 28(10), 885-918

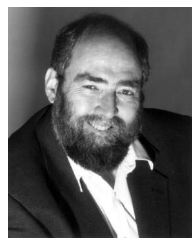
[Novak 84] J.D. Novak and D.B. Gowin, *Learning How To Learn*, New York: Cambridge University Press. 1984.

[Canas 98] The Giant: A Classroom Collaborator, Reichherzer, T. R., Cañas, A. J., Ford, K. M., Hayes, P. J., Workshop on Pedagogical Agents of the Fourth International Conference on Intelligent Tutoring Systems (ITS), San Antonio, 1998, pp. 83-86.

[Suthers 97] Suthers, D. & Jones, D. (1997). An architecture for intelligent collaborative educational systems. *AI-Ed 97, the 8th World Conference on Artificial Intelligence in Education,* Kobe Japan, August 20-22, 1997.

[Umeda 96] Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y. and Tomiyama, T.: "Supporting Conceptual Design based on the Function-Behavior-State Modeler," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 10, No. 4, (1996), pp. 275-288.

Author's Profile: Kenneth D. Forbus



Kenneth D. Forbus is a Professor of Computer Science and Education at Northwestern University. His research interests include qualitative reasoning, analogy and similarity, sketching and spatial reasoning, cognitive simulation, reasoning system design, articulate educational software, and the use of AI in computer gaming. He received his degrees from MIT (Ph.D. in 1984). He is a Fellow of the American Association for Artificial Intelligence and serves on the editorial boards of Artificial Intelligence, Cognitive Science, and the AAAI Press.