A qualitative modeling environment for middle-school students: A progress report

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

Karen Carney
Qualitative Reasoning Group
Northwestern University
1890 Maple Avenue
Evanston, IL, 60201, USA
harris@cs.nwu.edu

Robert Harris
Qualitative Reasoning Group
Northwestern University
1890 Maple Avenue
Evanston, IL, 60201, USA
harris@cs.nwu.edu

Bruce L. Sherin
School of Education
and Social Policy
Northwestern University
2115 N. Campus Drive
Evanston, IL, 60201, USA
bsherin@northwestern.edu

Abstract

Learning how to create, test, and revise models is a central skill in scientific reasoning. We argue that qualitative modeling provides an appropriate level of representation for helping middle-school students learn to become modelers. We describe a system we have created that uses visual representations to provide a student-friendly notation for creating qualitative models. This system is currently undergoing pilot testing in Chicago Public School classrooms, using curricula developed in collaboration with teachers.

Introduction

Modeling is a central skill in scientific reasoning. Learning to formulate, analyze, test, and revise models is a crucial aspect of understanding science, and critical to helping students become active, lifelong learners. Supporting students in articulating models of a domain, and refining them through experience, reflection, and discussion with peers and teachers, can lead to deeper, systematic understanding of science [6,23,26]. However, modeling is often treated as an art. Since modeling formalisms have traditionally been associated with creating mathematical models and deriving numerical results, they have been relatively inaccessible to younger students, such as middle school students. Moreover, many crucial aspects of models are traditionally not formalized, such as the conditions under which a model is applicable.

Qualitative reasoning formalisms provide the expressive power needed to capture the intuitive, causal notions of many human mental models [11,15]. This includes expressing aspects of modeling not handled by traditional environments, including conditions of applicability and other types of modeling knowledge. However, the predicate calculus based formalisms typically used in QR work provide a serious entry barrier to their use by children. The solution that we are exploring to this problem is to develop a *visual representation language*, based on concept maps, that provides a student-friendly way to express qualitative models. We have spent over a year doing design work, pencil and paper studies with students in classrooms, curriculum development, and software development, with the first formative tests of the software underway now. This paper summarizes our progress, focusing on the design of our modeling environment. We start by describing our approach in more detail. The design and software architecture of Vmodel (our working name for the system) is outlined next, focusing on the decomposition of the representations into *situation*, *causal*, and *evidence* maps. We then summarizes the collaborative work we have been doing with teachers to develop curricula and our classroom experiments. Finally, we close with a synopsis of our future plans

Approach

Why do we want students to become modelers? There are three important reasons. (1) 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 presenting their ideas to others for discussion and collaboration. Peer-peer questioning, discussion and justification of ideas has been shown to aid learning among young students [4]. (2) The process of modeling itself is valuable. It forces students to articulate relationships between entities and dependencies between their beliefs. This is important for both understanding the phenomenon being modeled but also in developing a broader understanding of complex, interrelated systems, an important goal in the AAAS standards [1]. The idea that a change in one variable may have far-reaching and unforeseen consequences is missing from many science curricula. The qualitative causal relationships developed by the QR community provide an appropriate level of expressive power for achieving this goal. (3) Modeling provides students with practice in using formal representations, a skill they will need for mastering mathematics and programming. It provides them with another source of experience in the creative, liberating power of a technical vocabulary. To use an analogy, a composer "debugging" a complex composition such as a symphony can quickly isolate entities (pitch, note length, chord structure) for analysis and change because they are expressed in a concise, agreed-upon technical vocabulary. Qualitative modeling, being grounded in everyday experience, can provide a bridge to more complex symbolic formalisms.

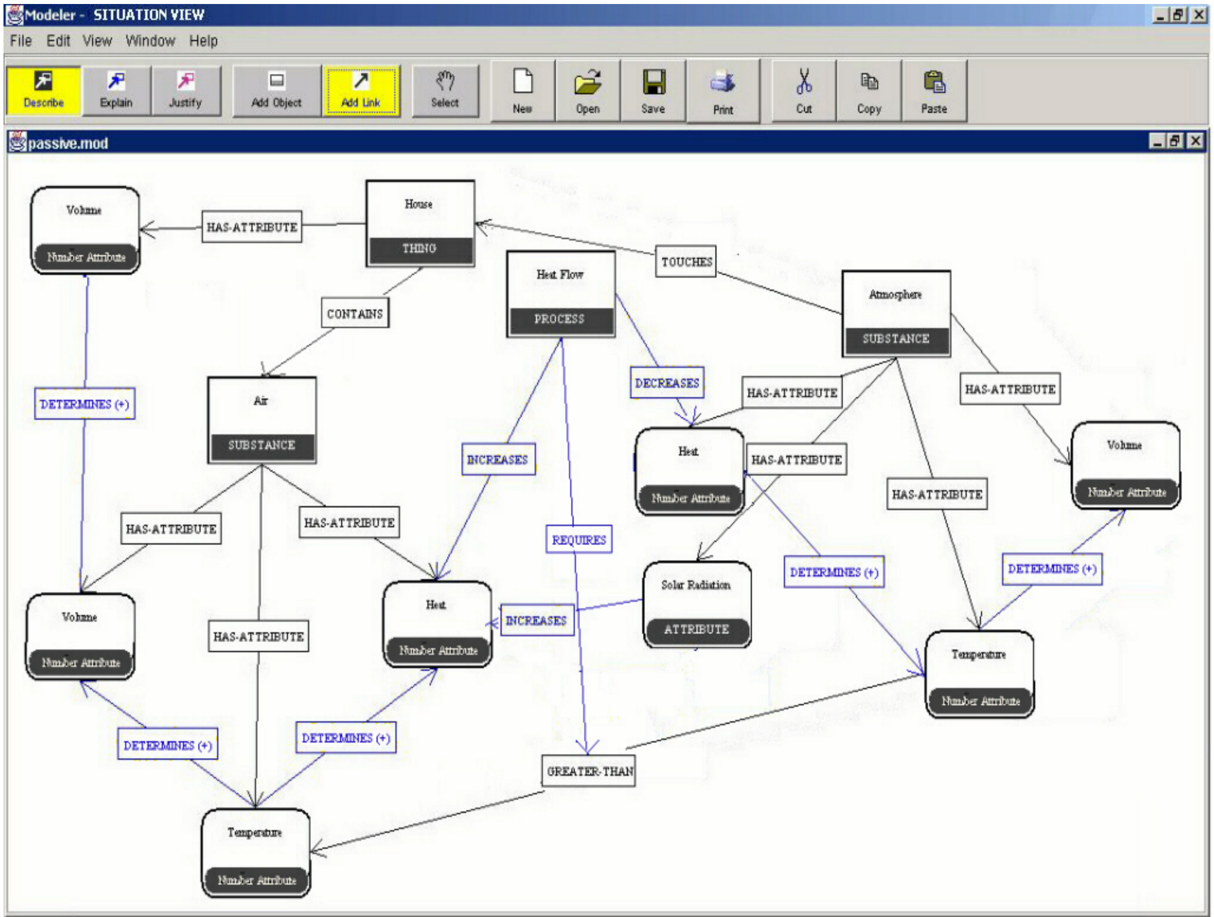
Given that we want to make students into modelers, how do we do it? Graphical external representations are powerful aids to thinking and learning. When coupled with computer support, graphical representations can provide powerful ways for people to communicate and collaborate even at a distance. Many graphical external representations have been created to aid students in articulating their understandings of phenomena. They can be grouped into three families: (1) *Concept Map notations*. Concept maps [24,5] 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 [17], Bond graphs, and software systems such as STELLA [18] and Model-It [19] provide graphical languages for expressing differential equations for continuous systems. (3) *Argumentation environments*. Belvedere [27], KIE [2], and the Collaboratory Notebook [8] 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 part of the knowledge and information involved in modeling, but none of them alone is sufficient. For example, concept maps adhere to minimal structural or semantic requirements. Although in theory they can be used to express anything, the lack of enforced or 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 [5,25]. 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 in detailed explorations. Even taken together, notations from these families, and curricula based on them that we are familiar with, 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 any 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 scientists do), as opposed to a series of ad hoc explanations concerning specific systems (as science taught in schools usually does) [26]. (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 numerical data plots as depicting behavior is an important skill, but providing all the data needed to run a numerical model can distract students from understanding the causal phenomena in the situation. In addition, the level of mathematics necessary to make a model of any reasonably complex phenomenon may be out of reach for a young student. Mathematics shifts the representation away from the basic level entities a student might naturally attend to. For example, a mathematical model necessitates students simultaneously representing all influences in a single equation or set of equations. This may be beyond the student. In addition, a student may not be able to easily interpret outcomes couched in mathematical terms or to debug wrong models.

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 [9] 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.

The design and architecture of Vmodel

We are creating a visual modeling tool for qualitative modeling that combines good ideas from concept maps, dynamical systems notations, and evidence maps into a qualitative modeling environment. The basic 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 `Number-Attribute`. 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 of whatever length 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.[1] This makes software coaching more feasible than in traditional concept maps. It also enables students to link their own propositions together, to articulate the reasons for their beliefs and record their argumentation. 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 reduces 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[2]. 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 will be organized into 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.

Our goal is for this environment to become as natural for modelers as word processors are to writers and spreadsheets are to accountants. That is, we are striving for something that can be used in all stages of modeling, from

[1] This restriction to binary relations entails no loss of expressivity for two reasons. First, as is well-known, higher-arity relationships can be expressed via reification. Second, the use of compositional relationships such as influences enables the assembly of more complex statements by underlying reasoning systems.

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

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 modeling. *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 and coaching.

Entities	Relationships
Basic Entities:	Property links:
Thing	Has-Attribute
Group-Thing	Does
Substance	Comparisons:
Time-Slice	Same as
Place	Is a kind of
Process	Greater/less than
Properties:	Equals
Attribute	Configural:
Number-attribute	Touches
State	Contains
Form	Part of
Observations:	Moves from/to
Observed change	Is changing.

Situation maps

Situation maps give a student the ability to describe the world and any observed behavior relevant to the model, providing a platform on which they can build their explanations of behavior. In the situation map, students describe the entities in the modeled world, as well as structural and configural relationships among them. The situation map enables students to articulate the entities and relationships that give rise to physical processes, which, in a well-developed model, provide the causal map.

Table 1 illustrates the ontology used in the situation map. 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-ATTRIBUTE relationship, whereas entities and the processes they participate in are indicated by DOES. Quantities are represented by NUMBER-ATTRIBUTEs, with ordinal information expressed via comparatives. Students are 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). Teachers and curriculum developers are allowed to extend the configural relationships vocabulary as necessary. Figure 2 illustrates a relational statement expressed using these conventions.

Figure 2: a simple relational statement in Vmodel.

Causal maps

The vocabulary for causal maps is drawn from Qualitative Process theory [14]. 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:

- `Linked`: The parameters are changing at the same time, and may be causally related, but directionality and nature of the causal connection, if any, is unknown.
- `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, in QP theory).
- `Determines(+)/determines(-)`: Indicates functional dependence between two parameters, i.e., the heat of something determines its temperature (qualitative proportionalities, in QP theory)

We believe that the fact that students can express partial relationships and make causal arguments about relations between entities in the world makes QP theory an ideal formalism for modeling for middle and high school aged students. Unlike the entity and configural relations vocabularies in the situation map, the vocabulary for the causal map is fixed, so that we can perform qualitative reasoning with the student's model.

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. The AAAS standards and other reform initiatives specifically call for more teaching about the epistemology of science in schools.

To implement evidence maps, we are adopting conventions from existing argumentation systems (including Belvedere, KIE, and WISE) to create another restricted concept mapping language. A major difference between standard argumentation systems and what we will embed in our software is that the grain size of our arguments (e.g., a node in a situation or causal map) needs to be much smaller than these environments typically use (e.g., an article or a web page). We are also investigating other external representations, such as tables and checklists, since such representations highlight different aspects of knowledge that may be useful in modeling [7]. For example, in evidence maps, checklists may be the best method for keeping track of what has yet to be explained. Similarly, specialized interfaces to describe the qualitative properties of behaviors that are to be explained may be necessary, to provide an easy way to enter graph-like data.

Although we are designing our representation to make the reasoning behind a model explicit, we think an important facility will be *annotations* on entities and links in models. Annotations enable students to express their understanding and thinking in free-form ways, e.g., "Jeremy and I disagreed about this". Encouraging students to add such information should encourage metacognition, aid recall of model-building experiences by anchoring them more concretely, and let them note information that they might later recast in terms of the explicit, visual notations. [1]

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 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 in the next section, we will 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 will be 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 will 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. One service that the software will provide as an encouragement for students to extend the Model Library is support for model formulation. That is, given a new situation, using automated modeling techniques from compositional modeling, the system will suggest relevant library elements that should be used.

As a student progresses, their Model Library will contain (aside from a relatively small set of primitive relationships and object types) 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

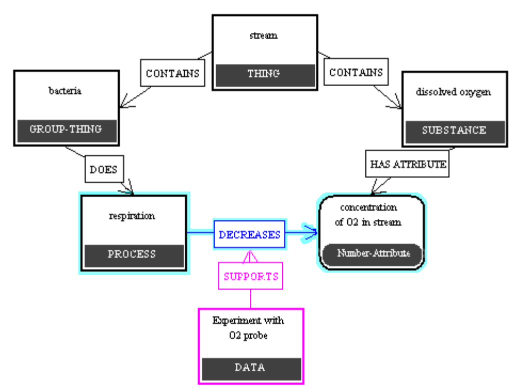


Figure 4: A Vmodel evidence map. The student highlights a portion of his or her model and supports it from data or theory in the evidence map.

existing model and specifying what aspects should be turned into connectors, and what information should be kept in the fragment. Figure 5 illustrates the process of creating a new type of entity for the Model Library.

The catalogs in the Model Library will be organized into trees for types of objects, relationships, and processes. Each tree will be organized via inclusion, e.g., a population of elk is a kind of population, which is a kind of group-

[1] The Belvedere researchers observed that students tended to use their environment only after their arguments were worked out (Suthers, personal communication). Teachers using Model-It in Chicago Public School classrooms have noted similar behavior (personal communication). Consequently, one of our design desiderata is providing as much support as we can for students to express their intermediate states of knowledge.

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.

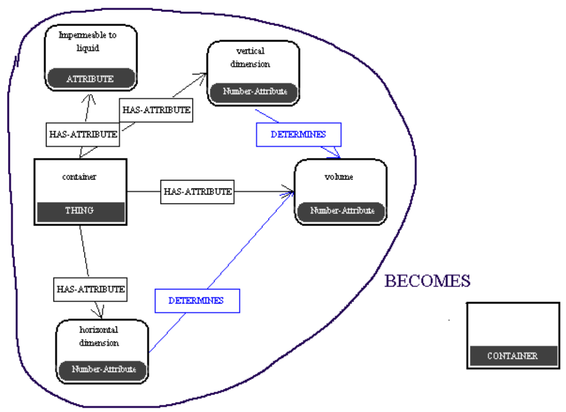


Figure 5: Creating a new type, container, from more primitive elements.

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. Consequently, we are using the *distributed coaching* model we first used with CyclePad [13], where the client software has built-in lightweight coaching services and more powerful coaching services are accessed remotely via electronic mail.

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 automatically as well.) We are building in simple coaching facilities, the modeling equivalents of spelling correctors and grammar checkers. Suppose for example that a student is using a description of heat flow from their Model Library, and they constrained the source and destination to be "Thermal Things". If they attempt to instantiate something as the source that isn't known to be a Thermal Thing, the on-board coach might ask whether or not the object in question is a Thermal Thing. Similarly, a proposed change in how one of their physical processes is represented can be automatically checked against their portfolio in the Model Library, to suggest how they might revisit old situations and reexamine them in the light of their new understanding.

For assessment and more sophisticated coaching, we are providing an email-based server, using our RoboTA agent colony [10], which provides an email "post office" and various housekeeping services for educational software agents. Students working in schools with network connections will be able to use an email facility built into the software. Unfortunately, many classrooms are not so endowed, and in those cases we resort to carrying a floppy or laptop back with us to get the data, or asking the teacher to email student work to us later using their personal accounts. Even if every classroom were networked, we would continue to use email, rather than web-based, coaching for scalability. One does not expect an instant reply to an email message, whereas a web server that does not respond instantly is annoying. Our collaborating teachers find this model reasonable because students will only be working a small part of any typical day on science activities, and that time will be spent in discussions as well as computer work.

We will be using analogical processing techniques [12] in the server-based coach. Providing feedback based on normative models is one example. (Normative models are authored using the same software, so that teachers and curriculum developers can add content without us being heavily involved.) Given that students can call entities and properties in their model anything they like, it would be useful to be able to use their terminology when appropriate. The ability of our structure-mapping engine (SME) to construct correspondences between representations should give us a "translation table" for such purposes. Furthermore, 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. A second use of analogical processing in coaching that we will explore is comparing student models within a classroom. By using SEQL [20] 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.

The tradeoff between client and server in this case is complicated. Based on our CyclePad experience, we would like to bundle as much coaching into the classroom software as possible: Speedy feedback is often preferred to promote learning. On the other hand, limiting on-board coaching encourages students to rely more on their own thinking and class discussion, instead of constantly seeking validation from the computer. We will start by offering new facilities through the server-based coach, and migrate them to the client software only after they have proved their utility. Two examples of such facilities are doing within-state qualitative reasoning to check whether their causal

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

map correctly predicts an observed change, and checking comparative analysis arguments.

Working with teachers and students

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*, a collaborative organization that involves researchers, teachers, and school administrators in developing and adapting materials for classroom settings. This joint development arrangement provides invaluable feedback and supports piloting materials and activities in a variety of urban schools

In collaboration with CPS teachers, over the last year we have been developing two curricula that are the initial settings for our modeling work. One curriculum concerns heat and temperature, in which students consider alternate energy resources for homes (cf. [21]). In collaboration with Marcia Linn's group at Berkeley, we are adding complementary simulation-based activities to their successful thermal curriculum [22]. These activities use self-explanatory simulators [16] as a way of allowing students to explore the various outcomes of their design choices in making a solar house. The other curriculum concerns ecosystems, using as a hook the creation of a life support system for a Mars colony. This will provide an arena for students to explore the requirements of life and how ecosystems work, using simulation experiments.

In parallel with the curriculum development work, we have been carrying out pilot studies in CPS classrooms to drive the visual notation design and the software design. In the early design phase, we used pencil and paper studies, which were very encouraging. We explored what relational vocabularies are most natural for students, by providing them with building blocks that have pre-built labels but also blank versions that they fill in. We are also beginning to characterize the kinds of difficulties students have with formalized representation systems, and catalogue ways in which students describe complex phenomena. Student trials using the current Vmodel software (i.e., without Model Library or coaching) are currently underway. Students are working with a simulation to explore the behavior of an ecosystem. They are then encouraged to build models in Vmodel to explain surprising or discrepant behavior that they encounter in their work with the simulation. The focus of this work is to understand what difficulties students may have with modeling using the Vmodel object ontology and relational vocabulary, prior to a larger-scale rollout.

Discussion

This paper summarizes the work we have done to date in creating a system 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 the representations needed 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. Based on our pencil and paper studies and interactions with teachers, we believe that this combination will be very useful for students.

We know of two projects that are very similar to ours in both spirit and approach. One is the UWF Quorum project [5,25], 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* found by James Lester. The second is "Betty's Brain" at Vanderbilt [3], 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.

The pilot studies underway with the current software will help us refine and revise our software. As we add coaching and the Model Library, we hope to create 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. 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 and Carlos Rodrigues.

References

1. American Association for the Advancement of Science (1989) Science for all Americans: a Project 2061 reports on literacy goals in science, mathematics, and technology. Washington, D.C., American Association for the Advancement of Science

2. Bell, P. (1997). Using argument representations to make thinking visible for individuals and groups. In R. Hall, N. Miyake, & N. Enyedy (Eds.), Proceedings of CSCL '97: The Second International Conference on Computer Support for CollaborativeLearning, (pp. 10-19). Toronto: University of Toronto Press.

3. 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.

4. Brown, A. L., Campione, J. C: Guided Discovery in a Community of Learners. *Classroom Lessons: Integrating Cognitive theory and classroom practice* Cambridge MA., MIT Press 229-270

5. 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.

6. 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.

7. Collins, A. and Ferguson, W. 1993. Epistemic Forms and Epistemic Games: Structures and Strategies to Guide Inquiry. *Educational Psychologist*, March, 1993.

8. 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.

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

10. Forbus, K., Everett, J., Ureel, L., Brokowski, M., Baher, J., and Kuehne, S. (1998). Distributed coaching for an intelligent learning environment. *Proceedings of the Twelfth International Workshop on Qualitative Physics* (QR98), Cape Cod, MA, USA.

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

12. 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.

13. 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.

14. Forbus, K. Qualitative Process theory. *Artificial Intelligence*, **24**, 1984.

15. Forbus, K. "Qualitative Reasoning". *CRC Handbook of Computer Science and Engineering*. CRC Press, 1996.

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

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

18. High Performance Systems, STELLA software, http://www.hps-inc.com/

19. 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.

20. 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.

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

22. 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

23. Minstrell, J. and Stimpson, V.C. (1990) A Teaching System for Diagnosing Student Conceptions and Prescribing Relevant Instruction. Paper prepared for AERA Session "Classroom, Perspectives on Conceptual Change Teaching." Boston.

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

25. 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.

26. Reif, F. and Larkin, J. H. (1991) Cognition in scientific and everyday domains: comparison and learning implications. *Journal of Research in Science Teaching*, 28. 773-760

27. 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.