Pianos Not Stereos:
Creating Computational Construction Kits

Mitchel Resnick (mres@media.mit.edu)
Amy Bruckman (asb@media.mit.edu)
Fred Martin (fredm@media.mit.edu)
MIT Media Laboratory
20 Ames Street
Cambridge, MA 02139
http://el.www.media.mit.edu/groups/el/
Published in *Interactions*, vol. 3, no. 6 (September/October 1996)

Introduction

Would you rather that your children learn to play the piano, or learn to play the stereo?

The stereo has many attractions: it is easier to play and it provides immediate access to a wide range of music. But "ease of use" should not be the only criterion. Playing the piano can be a much richer experience. By learning to play the piano, you can become a creator (not just a consumer) of music, expressing yourself musically in ever-more complex ways. As a result, you can develop a much deeper relationship with (and deeper understanding of) music.

So too with computers. In the field of educational technology, there has been too much emphasis on the equivalent of stereos and CDs, and not enough emphasis on computational pianos. In our research group at the MIT Media Lab, we are developing a new generation of "computational construction kits" that, like pianos, enable people to express themselves in ever-more complex ways, deepening their relationships with new domains of knowledge.

To guide the development of these computational construction kits, we are developing a theory of "constructional design." Whereas the traditional field of instructional design focuses on strategies and materials to help teachers instruct, our theory of constructional design focuses on strategies and materials to help students construct and learn. Constructional design is a type of meta-design: it involves the design of new tools and activities to support students in their own design activities. In short, constructional design involves designing for designers (Resnick, 1996b).

In recent years, a growing number of researchers and educators have argued that design projects provide rich opportunities for learning (e.g., Harel, 1991; Lehrer, 1993; Soloway, Guzdial, & Hay, 1994). In particular, Papert (1993) has argued for a "constructionist" approach to learning. There are many reasons for this interest in design-based learning. Design activities engage people as active participants, giving them a greater sense of control over (and personal involvement in) the learning process. Moreover, the things that people design (be they sand castles, computer programs, LEGO constructions, or musical compositions) serve as external shadows of the designer's internal mental models. These external creations provide an opportunity for people to reflect upon—and then revise and extend—their internal models of the world.

Of course, not all design experiences (nor all construction kits) are created equal. Some provide richer learning opportunities than others. What criteria should guide the design of new construction kits and activities? The concept of learning-by-doing has been around for a long time. But the literature on the subject tends to describe *specific activities* and gives little attention to the *general principles* governing what kinds of "doing" are most conducive to learning. From our experiences, we have developed two general principles to guide the design of new construction kits and activities. These constructional-design principles involve two different types of "connections":

- *Personal connections*. Construction kits and activities should connect to users' interests, passions, and experiences. The point is not simply to make the activities more "motivating" (though that, of course, is important). When activities involve objects and actions that are familiar, users can leverage their previous knowledge, connecting new ideas to their pre-existing intuitions. For artists

- *Epistemological connections*. Construction kits and activities should connect to important domains of knowledge—and, more significantly, encourage new ways of thinking (and even new ways of thinking about thinking). A well-designed construction For philosophers kit makes certain ideas and ways of thinking particularly salient, so that users are likely to connect with those ideas in a very natural way, in the process of designing and creating.

The challenge of constructional design—and it is a very significant challenge—is to create construction kits with both types of connections (e.g., Wilensky, 1993). Many learning materials and activities offer one type of connection, but not the other. In this article, we discuss three of our computational construction kits. In each case, we discuss how the kit aims to facilitate both personal and epistemological connections—and, as a result, support rich learning experiences.

Programmable Bricks

Traditional construction kits enable children to build structures and mechanisms, like castles and cars. The Programmable Brick adds a new level of construction, enabling children to build behaviors.

In Stop Drawing Dead Fish, Bret Victor uses this concept of behavior

The Programmable Brick is a tiny computer embedded inside a LEGO brick. Children can build Programmable Bricks directly into their LEGO constructions—then write programs to make their creations react, behave, and collect data. Some children have used Programmable Bricks to build autonomous LEGO "creatures" that mimic the behaviors of real animals (figure 1). Others have used Programmable Bricks to conduct new types of science experiments, investigating phenomena in their everyday lives. One 13-year-old boy connected a sensor to his leg and programmed a Programmable Brick (in his pocket) to keep track of the number of steps he took during a day. Another pair of students used Programmable Bricks to make a "smart room": when anyone entered the room, the Brick mechanically flipped the light switch; when the last person left the room, it turned off the light.

The Programmable Brick can be seen as a "very personal computer." About the size of a child's juice box, the Brick feels like the right size for a kid's computer; even a laptop seems too big in comparison. To use the Brick, children write programs on a personal computer, then download the programs to the Brick via a cable. After that, they can disconnect the Brick and take it (or put it) anywhere. The programs remain stored in the Brick. The Brick can control four motors at a time, receive inputs from six sensors, and communicate with other electronic devices via infrared communications.

Projects with the Programmable Brick often make strong personal connections to children's lives. Toys in general, and LEGO building materials in particular, are part of children's culture. When Programmable Bricks are added to the bin of LEGO building parts, computation becomes part of children's culture too.

At the same time, Programmable Bricks projects encourage strong epistemological connections. When students build and program robotic creatures, for example, they often wonder about the similarities and differences between animals and machines. Are their LEGO creatures like animals? Or like machines? They compare their robots' sensors to animal senses, and they discuss whether real animals have "programs" like their robots.

In one fifth-grade class, students created a LEGO dinosaur that was attracted to flashes of light, like one of the dinosaurs in Jurassic Park. This project had a clear connection to popular culture. But it also had a direct connection to scientific culture. To make the dinosaur move toward the light, the students needed to understand basic ideas about feedback and control. The program compared readings from the dinosaur's two light-sensor "eyes." If the dinosaur drifted too far to the left (i.e., more light in the right eye), the program made it veer back to the right; if the dinosaur went too far right (more light in the left eye), the program corrected it toward the left. This classic feedback strategy is typically not taught until university-level courses. But with the right tools, fifth graders were able to explore these ideas.

The Programmable Brick project is part of a larger Media Lab initiative known as Things That Think, and it relates to a field of research sometimes called "ubiquitous computing" (Weiser, 1991, 1993). The overarching goal in this research is to embed computational capabilities in everyday objects like furniture, shoes, and toys—mixing together "bits" and "atoms." The Programmable Brick fits within this initiative—but with an important twist. In our research, we are interested in Things That Think not because they might accomplish particular tasks more cheaply or easily or intelligently, but because they might enable people to think about things in new ways. That is, Things That Think are most interesting to us when they also act as Things To Think With. We believe that Programmable Bricks act in just that way: by enabling children to build their own Things That Think (like the light-seeking dinosaur), Programmable Bricks engage children in new types of thinking.

StarLogo

Whereas Programmable Bricks enable students to embed *computers in the world*, StarLogo enables them to construct *worlds in the computer*.

In particular, StarLogo is designed to help students model and explore the behaviors of *decentralized systems*—such as ant colonies, traffic jams, market economies, immune systems, and computer networks. In these systems, orderly patterns arise without centralized control. In ant colonies, for example, trail patterns are determined not by the dictates of the queen ant but by local interactions among the worker ants. In market economies, patterns arise from interactions among millions of buyers and sellers in distributed marketplaces.

Decentralized systems are important throughout the sciences and social sciences, but most people have difficulty understanding the workings of such systems. People seem to have strong attachments to centralized ways of thinking. When people see patterns in the world (like the foraging patterns of an ant colony), they generally assume that there is some type of centralized control (a queen ant). According to this way of thinking, a pattern can exist only if someone (or something) creates and orchestrates the pattern.

StarLogo is designed to help students make a fundamental epistemological shift, to move beyond the "centralized mindset" to more decentralized ways of thinking (Resnick, 1994, 1996a). With StarLogo, students construct and experiment with decentralized systems. They write simple rules for thousands of objects (e.g., artificial ants), then observe the patterns (e.g., colony-level foraging patterns) that arise from all of the interactions. By creating their own StarLogo models, students can build on personal connections. For example, two high-school students who had recently received their drivers' licenses used StarLogo to model to formation of traffic jams on the highway—a topic of great interest for them. They discovered (counter to their initial intuitions) how traffic jams can form through simple, decentralized interactions among cars, without any centralized cause like an accident, radar trap, or broken bridge.

Another high-school student, named Callie, used StarLogo to model the workings of a termite colony (figure 2). Callie had seen a television program that showed termites building intricate structures on the plains of Africa. She wondered how creatures as simple as termites could build such elaborate structures. She decided to program a colony of virtual termites to gather wood chips into a pile. At first, Callie tried to put one termite in charge, and programmed that termite to tell all of the other termites where to put the wood chips. But it is difficult to implement that type of centralized control in StarLogo. We discussed some of the drawbacks of a centralized leader: what would happen if the leader termite was killed? So Callie experimented with more decentralized approaches (more in line with the underlying structure of StarLogo) and found that the colony didn't need a leader after all. In her final model, each termite followed the same set of simple rules: wander randomly until you bump into a wood chip; pick up the wood chip; wander randomly until you bump into another wood chip; put down the wood chip you're carrying; then start over. This strategy uses only local sensory information, and a very simple control strategy, but the group as a whole accomplishes a sophisticated task.

Traditionally, these types of complex decentralized systems have been studied only at the university level, using differential equations and other advanced mathematical techniques. StarLogo enables much younger students to explore these systems—and to gain an understanding of the underlying ideas of self-organizing (Resnick, 1994) and probabilistic (Wilensky, 1993) behavior.

StarLogo makes these ideas accessible to younger students by providing them with a stronger personal connection to the underlying models. Traditional differential-equation approaches are "impersonal" in two ways. The first is obvious: they rely on abstract symbol manipulation. The second is more subtle: they deal in aggregate quantities. In the termite example, differential equations would describe how the density of wood chips evolves over time. There are now some very good computer modeling tools—such as Stella (Roberts et al., 1983) and Model-It (Jackson et al., 1996)—based on differential equations. These tools eliminate the need to manipulate symbols, focusing on more qualitative and graphical descriptions. But they still rely on aggregate quantities.

In StarLogo, by contrast, students think about the actions and interactions of individual objects. StarLogo is not simply a computerization of a traditional mathematical model; it supports what we call "computational models": models that wouldn't make sense without a computer. In the termite example, students think not about aggregate quantities but about individual termites and individual wood chips. They can imagine themselves as termites and think about what they might do. In this way, StarLogo enables learners to "dive into" the model, making a more personal connection. Future versions of StarLogo will enable users to zoom in and out, making it easier for users to shift back and forth in perspective from the individual level to the group level.

MOOSE Crossing

While StarLogo users typically build new worlds on their own, or in pairs, MOOSE Crossing provides a way for children to build virtual worlds together, as part of an online community.

In MOOSE Crossing, children not only "talk" with one another online, but also collaboratively construct (with words and computer programs) the virtual world in which they interact (Bruckman, 1994). MOOSE Crossing is similar to existing MUD environments (Curtis, 1992), but it includes a new programming language (MOOSE) and new client interface (MacMOOSE) designed to make it easier for kids to learn to program. For each object that kids create, they write a combination of text and computer code to describe the properties and behaviors of the object. For example, one twelve-year-old girl made a baby penguin that is always hungry. It responds differently when you offer it different kinds of food, and won't eat certain foods if it is on a diet. A nine-year-old girl made a magical room at the end of the rainbow—answer the riddle correctly and you can take the pot of gold. Children help one another with their projects, and share them with others excitedly. MOOSE Crossing places construction activities in a community context.

MOOSE Crossing is immediately appealing to many children, because it draws on their personal connection to computer games, to elements of popular culture, and to socializing with each other. The environment has the feel of a text-based adventure game (and historically has its roots in such games), but it opens up greater intellectual challenges: you not only can experience the world but also build it. Children often chose popular culture as the subject of their conversations, and an inspiration for their creations: for example, one afternoon, two twelve-year-old girls started talking about Star Trek and then decided to build themselves spaceships. Commercial culture is also a popular starting point for projects. One eleven-year-old girl first made a vacation resort called Paradise Island, next made a travel agency to sell people trips there, and lastly added a car rental agency. Several factors gave her a very personal connection to the project. It's important that she decided what she wanted to make: rather than being assigned a project, she chose one that was personally meaningful to her. Her entire participation in MOOSE Crossing was voluntary—children participate in their spare time as an after-school activity. She was especially motivated by a desire to share her creation with other children. On finishing Paradise Island, she immediately invited all of her online friends over for a swim. A successful project gives a child social capital within the community.

In each of these projects, children are doing creative writing and computer programming in their spare time for fun. MOOSE Crossing leverages from children's natural interests to engage them in these intellectually valuable activities. They establish a new relationship to reading, writing and programming, beginning to see them not just as something they are forced to do in school, but as expressive media through which they can make personally significant meanings. In other words, they establish a new epistemological relationship to these ways of understanding the world and expressing themselves.

MOOSE Crossing also establishes new connections between different ways of knowing that are often separated and isolated in school activities. Making a successful MOOSE object is equal parts creative writing and computer programming. For children who have a greater initial strength in one area, it helps them develop greater confidence and competence in the other. One nine-year-old girl who says she hates math and math-like activities loves programming on MOOSE Crossing because she sees it as a form of writing. Asked if she likes to write, she replied yes—in school she's writing stories about imaginary people; on MOOSE Crossing, she's writing programs. The only difference between these two kinds of writing is that "programming it everything has to be right so the thing you're making can work." She is bridging from her strong verbal skills to develop greater interest and skill in more analytic activities.

The children participating in MOOSE Crossing are mostly nine to thirteen years old; a few children are as young as seven. Adults may apply to be "rangers." While we originally expected rangers to help children with their projects, in practice it more often works the other way around. Children have much more time to devote to MOOSE Crossing, and generally understand how things work better than the adults. Assisting an adult with a technical question is a real thrill for many kids, and challenges some of their basic assumptions about learning. On MOOSE Crossing, everyone is playing, teaching, and learning all at the same time, rather like Seymour Papert's vision of activity in a "technological samba school" (Papert, 1980). Knowledge is not passed from teachers to students, but is developed by everyone through their activities and interactions with one another.

Emergent Learning Experiences

Programmable Bricks, StarLogo, and MOOSE Crossing are three very different types of computational construction kits. The first involves interaction with the physical world, the second involves the construction of virtual collaborations, the third involves collaboration on virtual constructions. What unites these three diverse environments is their attempt provide both personal and epistemological connections. Each of these kits connects to student interests and experiences, while also connecting to important intellectual ideas.

But the process of constructional design is not a simple matter of "programming in" the right types of connections. As students have used Programmable Bricks, StarLogo, and MOOSE Crossing, their learning experiences have been somewhat different than we (as developers) expected. This unpredictability is characteristic of constructional design. Developers of design-oriented learning environments need to adopt a relaxed sense of "control." Educational designers cannot (and should not) control exactly what (or when or how) students will learn. The point is not to make a precise blueprint. Rather, practitioners of constructional design can only create "spaces" of possible activities and experiences. What we can do as constructional designers is to try to make those spaces dense with personal and epistemological connections—making it more likely for learners to find regions that are both engaging and intellectually interesting.

In some ways, the design of a new learning environment is like the design of a StarLogo simulation. In creating StarLogo simulations, users write simple rules for individual objects, then observe the large-scale patterns that emerge. Users do not program the patterns directly. So too with constructional design. Developers of design-oriented learning environments can not "program" learning experiences directly. The challenge, instead, is to create frameworks from which strong connections—and rich learning experiences—are likely to emerge.

Acknowledgments

Many members of the Epistemology and Learning Group at the MIT Media Laboratory have contributed to the ideas and projects described in the paper. In particular, Brian Silverman, Randy Sargent, and Andy Begel played central roles in the development of the Programmable Brick and StarLogo. Many of the ideas in this paper were inspired by conversations with Seymour Papert, Alan Kay, and Mike Eisenberg. The National Science Foundation (9153719-MDR and 9358519-RED) and the LEGO Group have provided financial support for this research.

References

Bruckman, A. (1994). "MOOSE Crossing: Creating a Learning Culture." PhD dissertation proposal, MIT Media Lab. Available as ftp://ftp.media.mit.edu/pub/asb/papers/moose-crossing-proposal.{ps,rtf,txt}

Curtis, P. (1992). Mudding: Social Phenomena in Text-Based Virtual Realities. Proceedings of DIAC '92. Berkeley, CA.

Harel, I. (1991). *Children Designers*. Norwood, NJ: Ablex Publishing.

Jackson, S., Stratford, S., Krajcik, J., & Soloway, E. (1996). A Learner-Centered Tool for Students Building Models. *Communications of the ACM*, 39 (4): 48-49.

Lehrer, R. (1993). Authors of knowledge: Patterns of hypermedia design. In S.P. Lajoie & S.J. Derry (eds.), *Computers as Cognitive Tools*. Hillsdale, NJ: Lawrence Erlbaum.

Martin, F. (1994). Circuits to Control: Learning Engineering by Designing LEGO Robots. PhD dissertation. MIT Media Laboratory.

Martin, F. (1996). Ideal and real systems: A study of notions of control in undergraduates who design robots. In Y. Kafai and M. Resnick (eds.), *Constructionism in Practice*. Hillsdale, NJ: Lawrence Erlbaum.

Papert, S. (1993). *The Children's Machine*. New York: Basic Books.

Resnick, M. (1993). Behavior Construction Kits. Communications of the ACM, 36 (7): 64-71.

Resnick, M. (1994). *Turtles, Termites, and Traffic Jams*. Cambridge, MA: MIT Press.

Resnick, M. (1996a). Beyond the Centralized Mindset. *Journal of the Learning Sciences*, 5 (1): 1-22.

Resnick, M. (1996b). Towards a Practice of Constructional Design. In L. Schauble & R. Glaser (eds.), *Innovations in Learning: New Environments for Education*. Mahwah, NJ: Lawrence Erlbaum.

Roberts, N., Anderson, D., Deal, R., Garet, M., & Shaffer, W. (1983). *Introduction to Computer Simulation: A System Dynamics Modeling Approach*. Reading, MA: Addison-Wesley.

Soloway, E., Guzdial, M., & Hay, K. (1994). Learner-Centered Design. *Interactions*, 1(2): 36-48. April 1994.

Turkle, S. (1984). *The Second Self*. New York: Basic Books.

Turkle, S. (1986). "Computational Reticence: Why Women Fear The Intimate Machine." In C. Kramerae (ed), *Technology and Women's Voices*, NY: Pergamon Press.

Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*, 265 (3): 94-104.

Weiser, M. (1993). Some Computer Science Issues in Ubiquitous Computing. *Communications of the ACM*, 36 (7):75-84.

Wilensky, U. (1993). Connected Mathematics: Building Concrete Relationships with Mathematical Knowledge. PhD dissertation. Cambridge, MA: MIT Media Lab. n—and then revise and extend—their