Appeared in *Journal of the Learning Sciences* 9(3), 329-372, 2000.

Situated Support for Learning:
Storm's Weekend with Rachael

by
Amy Bruckman

Assistant Professor

College of Computing

Georgia Institute of Technology

Atlanta, GA 30332-0280

Tel: 404-894-9222

Email: asb@cc.gatech.edu

Submitted: April 22nd, 1998

Revised: January 7th, 2000

Situated Support for Learning:
Storm's Weekend with Rachael

Abstract

While much attention has been paid to the *content* of support for learning, less attention has been given to its *context*. This paper introduces the notion of "*situated support*," and argues that the identity of the source of support and the connectedness of that support to other elements of the learning environment are of primary importance. MOOSE Crossing is a text-based virtual reality environment (or "MUD") designed to be a constructionist learning environment for children eight to thirteen years of age. A microanalysis is presented of the situated nature of support for learning on MOOSE Crossing over the course of one weekend where a twelve-year-old girl learned to write simple computer programs.

Situated Support for Learning:

Storm's Weekend with Rachael

1. Introduction

1.1 Storm's Weekend with Rachael

One Friday afternoon in April of 1996, I accepted the MOOSE Crossing application of a new member, a twelve-year-old girl who chose the character name Storm[1], and then I left town for the weekend. MOOSE Crossing is a text-based virtual world (or "MUD") designed to be a constructionist learning environment for children (Bruckman 1997, Bruckman 1998). Children on MOOSE Crossing are creating a virtual world out of words, making magical places and creatures that have behaviors. In the process, they are improving their reading and creative writing skills, and learning how to write computer programs.

I usually log on periodically over weekends, but this particular weekend I went to Maine and was offline. When I returned on Sunday, I was surprised to learn that Storm now knew the basics of how to program. She had had limited previous experience—she had once tried Logo in school. Over the weekend, she made Jasper (a frog you can hug), Callie (a cat who purrs when you tickle her and responds wryly when you address her as "cat" rather than by name), and a catnip mouse for Callie. She also built and described three homes, and collaborated with Rachael (girl, age 13) on an extension to Rachael's castle, complete with a dead princess on the floor who had died for love. Storm and Rachael spent most of the weekend together, talking and helping one another with their projects.

I've told this story many times, and people often shake their heads with disbelief, grinning. But what is there about the story that is remarkable? It's certainly not how we stereotypically imagine two children spending a holiday weekend. It's hard for many people to imagine any "real people" having meaningful control over computational media, much less girls. What was it that interested them so completely that they would spend most of the weekend absorbed in it? What were they doing and how did they learn to do it? What kinds of support were available to them?

This paper introduces the notion of "situated support" for learning. Research into "intelligent tutoring systems" focuses on modeling the learner and offering support custom-tailored to their needs (Anderson, Corbett et al. 1995). Research into "scaffolding" focuses on getting support to the learner at the right level, and fading the support offered when it is no longer necessary (Guzdial 1994; Rogoff 1994). Much detailed research has been done on types of scaffolding. Jackson, Krajcik, and Soloway identify basic types: supportive, reflective, and intrinsic (Jackson, Krajcik et al. 1998). Within the category of reflective scaffolding, Elizabeth Davis examines prompts directed at engaging reflection, and compares the relative merits of generic and directed prompts (Davis 1998). Building on this research, I add another dimension to the analysis: the origin of that support. This paper argues that the utility of support is relative to its origin, whether computational or human. Certainly the *content* of the help you receive matters, but the *context* in which that support is situated is also of great importance. It matters who or what the support is from, and how the support is connected to daily activities and to other potential sources of support.

"Context" is always changing: it includes not just durable features of the environment, but also more ephemeral social phenomena. While we can not

---

[1]The children's real and character names have been changed.

comprehensively specify the context for a particular instance of support for learning, we can compare what we do know about multiple such instances to try to understand the context's most salient features. I conclude that support for learning is more valuable when it is:

- From a source (either human or computational) with whom the learner has a positive personal relationship,
- Ubiquitously available,
- Richly connected to other sources of support, and
- Richly connected to every-day activities.

1.2 Help from Whom?

I approach the notion of "support for learning" from a social constructivist perspective. Newman, Griffith and Cole comment that "when people with different goals, roles, and resources interact, the differences in interpretation provide occasions for the construction of new knowledge" (Newman, Griffin et al. 1989). This approach to learning has dual roots in the work of Jean Piaget (Piaget 1929) and Lev Vygotsky (Vygotsky 1978).

Getting guidance from another person is a complex process that exists in the context of networks of social relations. Donald Schön documents the subtleties of the relationship between masters and apprentices, with particular focus on professional education (Schön 1987). Schön pays careful attention to the unequal power relationships in apprenticeship situations. Jean Lave and Etienne Wenger also examine the relationships between masters and apprentices, with greater emphasis on new members of a community of practice being gradually introduced to a craft through the process of "legitimate peripheral participation" (Lave and Wegner 1991). Collins, Brown and Newman refine the notion of traditional

apprenticeship to one of "cognitive apprenticeship" (Collins, Brown et al. 1989). In contrast, Aaron Falbel describes an unusual "free school" in Denmark, Friskolen 70, where the power relationships are more nearly equal. Learning at Friskolen 70 is self-motivated and peer-supported, as Falbel describes in this vignette:

> Maria (11) and Johanne (9) are using the wood-burning set in the workshop to make Christmas presents for their friends and relatives. Maria has just completed a small, wooden tic-tac-toe game for a cousin of hers in Sweden. The X's and O's are made out of wooden pieces that fit neatly into a finely crafted playing board. She is etching an inscription to the recipient of the gift on the back of the playing board when little Clara (6), who just started at the school at the beginning of the month, wanders by and is drawn into the scene. She watches Maria etch a floral design under her inscription and is fascinated by the strangeness of this smoldering writing instrument. Maria notices the presence of Clara and displays for her the finished product. Clara turns over the board and feels the grooves of the inscription with her fingers. Maria asks her, "Can you read?" Clara shyly shakes her head no. Maria smiles and says "Come," and she motions for Clara to come sit on her lap. Then, very slowly and sweetly, Maria sounds out the words as Clara guides her fingers over the dark-brown letters. Clara is totally absorbed: her face conveys an expression of rapt concentration, her mouth partly open, and her cheek leaning against Maria's arm. The entire episode lasts not much more than a minute. It is so effortless, natural, and unself-conscious that to call it "peer tutoring" would be to debase the beauty of the situation. (Falbel 1989)

Analyzing this anecdote, one could conclude that appropriate content was delivered to Clara at a developmentally appropriate time. However, the context in which that support was delivered is as important as the content. This was not just a reading lesson from a book, a computer, an adult authority figure, or a stranger; it was a reading lesson from an older child whom Clara apparently admires. Similarly, we will see that over the course of that April weekend, Storm received help not just from any source, but from a girl six months older with whom she has many common interests. Discussing the role of peers in learning, Johnson and Johnson write, "In their interactions with peers, children and adolescents directly learn attitudes, values, skills, and information unobtainable from adults. In their interactions with each other, children and adolescents imitate each other's behavior

and identify with friends possessing admired competencies. Through providing models, reinforcement, and direct learning, peers shape a wide variety of social behaviors, attitudes, and perspectives" (Johnson and Johnson 1987).

Falbel takes the radical stance that power relationships should be equalized as much as possible—the artificial authority of status and age should be replaced by the natural authority of greater knowledge and experience (Falbel 1989). In contrast, Barbara Rogoff argues that we need to seek a balance between adult-run and child-run models of learning (Rogoff 1994). Regardless of which of these stances we adopt, it's clear that peer tutoring is a powerful educational technique that is underused in our educational system (Johnson and Johnson 1987).

Computer-Supported Collaborative Learning (CSCL) systems (Koschmann 1996) have affordances that make learning from peers practical. A virtual community (Rheingold 1993) can make a group of peers with shared interests and values conveniently available from home or school at whatever hour they are needed.

Not all help comes directly from a person. However, it's worth noting that written and computational artifacts that might serve as support for learning all ultimately have a human origin. If the identity of that human origin is known to the learner, then the learner's relationship to the artifact is affected by the learner's relationship to that author.

1.3 Ubiquitous, Connected Help

The ultimate human source of support is just one component of its context. Other important factors include where the help is found, and whether it is isolated or connected to other sources of help. Help is more valuable when it is everywhere all around the learner (ubiquitous), and connected both to other sources of help and to the learner's every-day interests and activities. A particularly good example of ubiquitous, connected help is HTML coding on the World Wide Web. On the

Web, you can usually view the HTML source for any document. When you want to know how to do something yourself, you can remember something you saw that uses that technique, and look at the HTML source code. As you are going about your daily business using the Web, you are developing an ever richer vocabulary of models you can refer back to. Most software products come with detailed instruction manuals and sometimes libraries of examples; however, it is a truism that users tend not to take the time to read them. You can not learn from an example you have not seen. On the Web, a user's knowledge of available examples increases continually in the course of normal activity.

Similar to the Web, every object on MOOSE Crossing is an example to learn from. A learner in a virtual place can look at every object present—objects, other characters, the room itself—and use those as models to learn from. The community of MOOSE Crossing is of course multiple orders of magnitude smaller than that of the World Wide Web. It's akin to comparing the impact of new architectural forms in a small town and a big city. An innovative creation in a small town will be noticed by a high percentage of the population, and is likely to have a strong influence on the town in the future. An innovative creation in a city might be seen by more people total, but by a lower percentage of the population, and is likely to have a less discernible impact on future designs in the area.

Since the set of examples to learn from is the set of all things everyone in the community has ever made, those examples reflect the interests of the community. That set of examples is not static; it grows and evolves as the community grows. A single, centralized author of a set of examples could never reflect the community's interests and needs as well as the decentralized process of using everyone's creations as examples to learn from. The help available to HTML coders on the Web and MOOSE programmers on MOOSE Crossing is situated help.

1.4 Methodology

In this article, I describe in detail what took place between Storm and Rachael over that weekend. Everything typed on MOOSE Crossing is recorded, with written informed consent from both parents and children. Over that weekend alone, 3.7 Mb of data was recorded of their experiences. This data includes everything each girl typed and saw on the screen—everything they did, and everything they said to one another online. The research methodology used in this study bears little if any resemblance to the physical sciences, but instead is inspired primarily by cultural anthropology. By examining what took place in extended detail, I am attempting to present something like what Clifford Geertz calls a "thick description" (Geertz 1973). A "thick description" allows us to distinguish the difference, in Geertz's example, between a contraction of the eyelid and a "wink"—detailed knowledge of the context of the action allows one to move from factual observation to interpretation. While an analysis of primarily online data could never be as "thick" as the kind of extended face-to-face study that Geertz proposes, the methodology of this study is inspired by this sort of anthropological approach.

Methodologically, studying this online medium has interesting properties: we have in some respects more and in others less data than is available in a traditional ethnography. On the one hand, the subjects could communicate only through the computer, and every keystroke was recorded. Their communication was reduced to bits of information on a computer network, and we have a comprehensive, objective record of that communication. On the other hand, we can not see their faces or hear comments made out loud to family members. Moving from factual quotation of the written record to understanding the significance of what took place is necessarily an interpretive act. Where the written record is insufficient to move from fact to interpretation, that record has been supplemented with face to face interviews and email exchanges with the subjects, as will be discussed below.

More data is included here than is strictly necessary to make this article's central point about the significance of situated help. Extra detail helps to form a complete narrative of the weekend's events, gives the reader a more complete picture of the nature of the learning environment, and leaves open the possibility that the reader may form his or her own interpretation of the events that took place.

Seven months after this weekend took place, I sent Storm and Rachael copies of an earlier version of this article to read and comment on. Social science has in recent years increasingly rethought the issue of ethnographic authority. Henry Jenkins writes that "The newer ethnography offers accounts in which participation is as important as observation, the boundary between ethnographer and community dissolves, and community members may actively challenge the account offered of their experience" (Jenkins 1992). I chose to give the girls an opportunity to respond to my account of their experiences. After they had each read the article, I invited them to come to the Media Lab[2] to discuss it with me, and to meet one another for the first time. It was fortunate that Rachael lives in the greater Boston area, and Storm lives an approximately ninety-minute drive away. (I'm grateful to Storm's parents for taking the time to make the trip.) I interviewed them both individually and together about their experiences that weekend and on MOOSE Crossing in general, and recorded those interviews on audio tape. A semi-structured, clinical interview style was used (Seidman 1998). The girls' parents were also interviewed briefly. Over all, Storm and Rachael enthusiastically concurred with my account. As I had hoped, the interviews helped me to clarify what they each were thinking and feeling over the course of the weekend. As a bonus, it also gave me an opportunity to ask the girls in what ways meeting face to face was different from meeting online.

---

[2] At the time this research was conducted, I was a graduate student working with Mitchel Resnick in the Epistemology and Learning Group at the MIT Media Lab in Cambridge, Massachusetts.

This case study examines what took place over that weekend in detail, with particular emphasis on the situated nature of the available support for learning. This particular weekend was not selected randomly. It was selected as a positive example of a successful learning experience. By analyzing the factors that contributed to that success, we can also work to understand what might go wrong when those factors are not present. Storm's experiences were more successful than average, but they are not unique—many other such positive instances could be studied. My method here is not to generalize across cases, but to do a micro-analysis of one interesting case to understand it in fine detail.

1.5 Setting the Scene: The World of MOOSE Crossing

MOOSE Crossing is a text-based virtual reality environment or "MUD." For historical reasons, "MUD" stands for "Multi-User Dungeon" (Bartle 1990). The first MUDs, developed in the late 1970s, were networked multi-player dungeons and dragons games. In 1989, a graduate student at Carnegie Mellon University named James Aspnes decided to see what would happen if he removed the monsters and the magic swords from the game, and instead allowed users to extend the virtual world. In most MUDs up until this time, extending the virtual world was a privilege reserved for either the owners of the computer the game ran on, or those who had dedicated hundreds of hours of playing time and succeeded in winning the game by completing all of its quests. In allowing everyone extend the virtual world, Aspnes' primary goal was to make the game more fun; however, in the process he created an unusual new kind of constructionist learning environment (Aspnes 1992).

The "world" of a MUD is a kind of database of objects. Each object has a name, a textual description, a location, and other properties. If you "look" at an object, you "see" its description. For example:

```
→ look at Rachael

A girl with brown hair and green eyes. On her head is a silver
headband with silver strands. At the end of each strand is a
silver ball. Around her neck is a silver chain.
```

Each person has a character object or "avatar." The three basic kinds of objects are people, places, and things. Objects also may have programs that allow you to do things with them. For example, you might want to pet Rachael's pig Rally:

```
→ pet rally

You pet rally.
Rally squeals happily.
```

When you examine any object (by typing "examine <object name>"), the system gives you a list of available commands. In addition to being able to pet Rally, you can also hug it, kiss it, kick it, etc. The programs that give Rally these capabilities were written by Rachael. MOOSE Crossing includes a new programming language and environment which I designed to make it easier for children to learn to program (Bruckman 1997).

Two people connected to the virtual world at the same time in the same virtual room can "talk" to one another. If I type "say hi," I see:

You say, "hi"

Everyone else in the same room in the virtual world sees:

```
Amy says, "hi"
```

If I type, "emote smiles," everyone sees:

```
Amy smiles
```

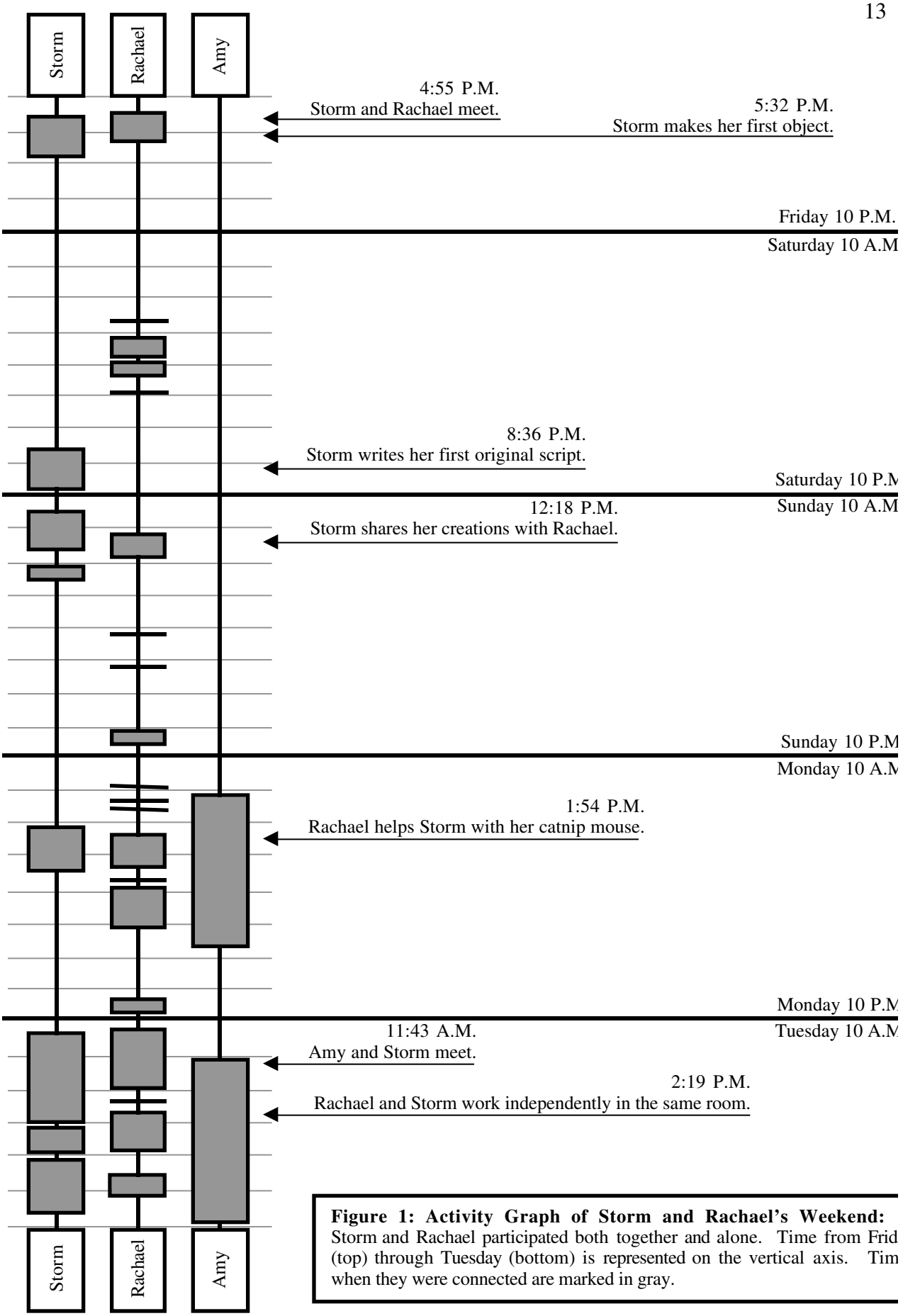

Figure 1: Activity Graph of Storm and Rachael's Weekend: Storm and Rachael participated both together and alone. Time from Frid (top) through Tuesday (bottom) is represented on the vertical axis. Tim when they were connected are marked in gray.

The "say" and "emote" commands allow the user to communicate with everyone in the same virtual room. The "whisper" command lets you communicate privately with just one person in the room. The "page" command lets you communicate with people connected at the same time but located in a different room in the virtual world. A built-in mail system lets users communicate with others who are members of MOOSE Crossing, but don't happen to be connected at the same time.

Conversations in such environments are often multi-threaded: while one person types an answer to a question, another may comment on a new topic. Both topics then may continue to be discussed in an A/B/A/B conversational structure. To add to the complexity, text "spoken" by people may be interleaved with text automatically generated by pets like Rally and other objects. Linguist Lynn Cherny describes the unusual features of MUD discourse in detail (Cherny 1995; Cherny 1999). While records of such conversations may seem complex to the uninitiated, most participants learn to read them with ease in a relatively short time (Negretti 1999). Transcripts in this paper have been annotated to aid the reader.

At the time Storm met Rachael, approximately 100 children and teenagers had participated in MOOSE Crossing. At the time of this writing in 1999, that has grown to roughly 400. Children's degree of participation ranges from those who tried only a few commands and never returned, to Rachael who participated actively for years and has typed more than 150,000 commands. Storm has typed a total of roughly 10,000 commands. The median number of commands typed during their total MOOSE Crossing participation for all children is 375, and the mean is 2440.

Over that April weekend, Storm used a variety of learning strategies and sources of support. In this transcript she makes use of:

- System resources:
  - the introductory message sent to all new members with their passwords
  - the online help system
  - online tutorials
- Learning strategies:
  - trial and error
  - using others' projects as models
- Social support:
  - unsolicited support from peers
  - solicited support from peers

In analyzing these various sources of support, this paper will focus not on the support's *content*, but on its *context*. Storm's learning supports were:

- From friendly sources (i.e., sources that she has a generally positive personal relationship with),
- Ubiquitously available,
- Richly connected to other sources of support, and
- Richly connected to routine activities.

2. Friday: Support From a Friendly Source

Rachael, who at this time had just turned 13 the previous week, is one of MOOSE Crossing's most dedicated and accomplished regulars. (She first connected on January 20th, 1996, and participated often several hours per day and on average roughly six days per week in 1996 and five days per week in 1997). She was the

first home-schooled student to join. Being home-schooled gives her greater time to devote to MOOSE Crossing, and also greater need for social contact.[3]

A graph of Storm and Rachael's activity over the weekend appears in Figure 1. An hour and a half after I accepted Storm's application, Rachael checks the list of all members, notices that there is a new member who hasn't logged on yet, and sends her this mail:

```
Message 1 on Storm:
From:     Rachael
To:       Storm
Subject:  hi

Dear Storm,
Hi! My name is Rachael. Who are you? I am thirteen years old
and I am female. I have been on moose crossing scince[4] january
and whould love to be your friend.  The best times to go on
moose crossing are on mondays and fridays afterschool.
Rachael
```

At a little before 5 PM, Storm connects for the first time. Her connection message tells her she has mail, so she checks "help mail" and figures out how to read her message from Rachael. She told me in a later interview that she learned how to use the online help system and a few other basic commands from the introductory message mailed to all new members with their passwords. She learned about the mail system because she wanted to read her mail from Rachael—interactions with other children were an integral part of her explorations with the system from the very first command typed.

---

[3]Over time, MOOSE Crossing has become particularly popular with home-schoolers. David Mankins, organizer of the home-ed mailing list for home-schooling parents, comments that parents chose to home school their children for a variety of reasons, but fall into three primary partially overlapping groups: those with strong religious views, those who believe in a variety of "progressive" educational theories, and those just dissatisfied with their local school system (Mankins 1999). The open-ended, self-directed educational philosophy behind MOOSE Crossing is consistent with the educational philosophy of the progressive wing of the home-schooling movement. Whatever the reason a child is home schooled, MOOSE Crossing provides much-needed social contact with peers.

[4]The children's writing has been left unedited. Spelling and grammatical errors have not been corrected.

A moment later, Rachael notices that Storm has connected, and pages her "May I join you?" ("Paging" is a way of communicating with someone who is not in the same room in the virtual world.) Rachael waits impatiently for Storm to respond, repeatedly checking 'who' (a command that tells you who is logged on, how long they've been connected, and how long they've been idle). She also repeatedly checks Storm's last commands. It's possible to see all the commands someone has typed recently. We added this feature to MOOSE Crossing to make it easier to help others figure out what they're doing wrong when things aren't working right. When you look at someone's last commands, the system tells the person that you have looked. Thirty seconds later, Storm pages Rachael "Yes" and Rachael joins her, moving to be in the same room as Storm in the virtual world:

```
Rachael beams in.
Rally says, 'Greetings Clover'
Rally arrives, following Rachael.[5]
Clover arrives, following Rachael.
Rachael says, 'hi'
Storm says, 'hello, all'
Rachael smiles.
Rachael says, 'Rally and Clover are my pets.'
Storm smile
Rachael says, 'how old are you?'
Storm says, '12'
Rachael nods.
```

Rachael looks at Storm. Every object and character can have a description that you use the "look" command to "see." At this time, Storm's description is still blank.

```
Rachael says, 'Are you at the media lab or somewhere else?'
Storm says, 'this is the first time I've been here!:)'
Rachael says, 'I mean in real life where are you? I'm at my
   house.'
Storm says, 'same here'
Storm says, ' anyone here like star trek?'
```

[5]For technical reasons, events on MOOSE Crossing sometimes happen in an unusual order—Rachael's pet Rally says hello before he is announced as arriving in the room.

```
Rachael says, 'Many people go to the media lab, in MIT, to do
   moose crossing. I was just wondering if you where. Yes, I
   do.'
Storm says, 'i'm a trekker, heart and soul!:)'
Rachael giggles.
Rachael says, 'austina/kristina[6] likes it too.'
Storm says, 'whoohoo!'
```

The expression "whoohoo!" became popular around this time in the mid-1990s, and is taken from the television show *The Simpsons*. Homer Simpson says it when he's happy. Popular culture, particularly science fiction television, immediately gives these two girls something to talk about. Their mutual interest also identifies them to each other as being part of an unusual sub-group of teenagers: girls who like science fiction. While many cultural critics scorn popular culture as a negative influence (Postman 1985), in the right environment it can be a rich source of raw materials for personal expression (Jenkins 1992). Popular culture is an integral part of Storm's support for learning in this environment: it helps to motivate and sustain her interest, and gives her stronger connections to her primary human support (Rachael) and the projects she is working on.

```
Rachael says, 'which generation do you like best?'
Rachael says, 'I like the TNG'
Storm says, 'trekkers will take over the world! I like the next
   generation
Rachael smiles. "Mabye so!"
Rachael says, 'same here, the next generation. Isn't a shame
   that it stopped?'
Storm says, 'oh, man yes! bummer!\'
Rachael says, 'Whould you like any help?'
Rachael says, 'I hear that the next TNG (next generation) movie
   is about the borg.'
```

Rachael guesses that Storm doesn't understand that the abbreviation "TNG" stands for "The Next Generation," and here finds a way to let her know without making a big deal out of it. From the start, the girls are attentive to one another's

---

[6]The characters Austina and Kristina are played by the same person, Austina DeBonte. At this time, Austina was an undergraduate research assistant working on the MOOSE Crossing project.

level of understanding and use conversation to work towards "convergent conceptual change" (Roschelle 1996). While this initial example is relatively trivial, this is the beginning of a process that later in the weekend will extend to more substantive concepts about programming and creative writing.

```
Storm says, 'oops, didn't mean the \. no, i'm figuring this
   out! thanks!'
Storm says, 'well, the movie is about the two generations
   meeting. You should see it!'
Rachael says, 'that's fine. Just askin'. But if you want any
   don't be afraid to ask.'
```

Storm hasn't yet had time to explore or get confused about anything yet—Rachael contacted her within moments of her arrival. Rachael's immediate offer of help is a bit premature. While Storm does not immediately avail herself of that help, she now knows that help is available when she needs it.

Rachael and Storm's conversations about help with MOOSE Crossing and Star Trek become interwoven. This is typical of MUD exchanges. In face-to-face discourse, turn-taking conventions help to keep a conversation focused. In MUDs, while one person is responding to a previous point, another may introduce a new one. Typing in parallel often leads to many-threaded conversations (Cherny 1995; Cherny 1999).

```
Rachael says, 'Oh, I saw it. I meant the one they are making
   now.'
Storm says, 'anyone like monty python?'
Rachael says, 'Yes, but I never get a chance to see it.'
Rachael says, 'Have you heard of Babylon Five?'
```

Rachael explained earlier that Rally and Clover are her pets; however, from her use of "anyone," it's not clear if Storm yet understands that it's only really Rachael there. At this point, Storm looks at Rachael. She sees:

```
A girl with brown hair and green eyes. On her head is a sliver
headband with silver strands. At the end of each strand is a
silver ball. Around her neck is a silver chain.
She is awake and looks alert.
Carrying:
Franky                                        Rachael's Bean
Rachael is wearing a tye die shirt and overalls.
Rachael smiles.
```

Rachael had written a special script (called a "look_self" script) which is run every time someone looks at her. She is now notified "Storm just looked at you." The program also causes her automatically to smile.

Rachael next looks at herself, to see what Storm has seen. Storm tries to look at Franky and Rachael's bean, not realizing she can't see objects that someone else is holding. Rachael audits Storm (a command which shows you all the objects a person owns), and sees that she still owns nothing.

```
Rachael says, 'Do you like the way I look?'
Storm says, ' you look bea-u-ti-ful!'
Rachael says, 'thanks.'
Rachael says, 'Well, have you heard of babylon five?'
Rachael says, 'It is another science fiction TV show.'
Rachael hugs Rally.
Rally squeals happily.
```

Rachael here gets a bit fidgety, typing "who" a couple times and hugging her pig Rally. Storm is busy describing herself. Rachael guesses as much, and looks at Storm immediately after the description is finished. She sees "you see a tall, black haired, white - skinned girl, wearing all black. she is wearing lots of silver jewelry."

```
Storm says, 'never seen it'
Rachael says, 'Most people haven't. You look nice!'
```

[Storm here looks at Rally, and sees "A small pink piggy."]
`Storm says, 'why thank you!'`

Storm gets immediate positive feedback. The feedback has value on several levels. First, it's a compliment to her technical ability—she was able to figure out how to describe herself. Second, it's a compliment to her writing ability. Third, being told you look nice is generally a mark of social acceptance. Given the predominance of issues of appearance in the culture of teenage girls, being told that your virtual self looks nice potentially has deep resonances. It's important to note that this positive feedback is situated: it is coming from a peer who is rapidly becoming a new friend. The exact same feedback would not have the same meaning if it were issued automatically by a computer program. The source of the feedback matters, not just its content.

Feedback often plays a key role in the growth of understanding through inquiry. Michele Wisnudel Spitulnik writes:

> The "teaching for understanding" and cognitive apprenticeship frameworks detail constructivist learning environments that support students in building understanding through inquiry (Collins, Brown et al. 1989; Perkins 1996). These frameworks suggest teachers should adopt facilitating and guiding strategies to support student inquiry rather than transmitting information to students. Examples of these instructional strategies include 1) making expectations/goals explicit (Perkins 1996); 2) providing feedback (Collins, Brown et al. 1989; Perkins 1996); 3) modeling desired cognitive processes (Collins, Brown et al. 1989); and 4) using examples or analogies. (Spitulnik 1999)

Although there is no adult teacher guiding Storm, over the course of this case study she will benefit from three of the four facilitating strategies that Spitulnik highlights. Here she receives feedback; later Rachael will model what Storm can accomplish on MOOSE Crossing, and provide examples.

The one strategy from Spitulnik's list that is missing is making goals explicit. This is a voluntary free-time activity for Storm, and nothing is expected of her. (Storm participates from home; some children use MOOSE Crossing from school.

See (Bruckman and DeBonte 1997).) Self-selected goals have both advantages and disadvantages. Seymour Papert quotes Albert Einstein in saying that "love is a better master than duty" (Papert 1985). Field observations repeatedly document that many children do love what they learn on MOOSE Crossing, and those children are often not the same children who typically do well in school. However, many other children often fail to challenge themselves and don't reach the level of accomplishment of Storm or Rachael. Annual MOOSE Crossing community events like the Holiday Pet Show and Spring House and Garden Show were designed to help children to set goals for themselves. Projects are nominated for awards in a variety of categories such as best written, best scripted, funniest, etc. Members vote for their favorites, and awards are presented at a public ceremony. Children have been observed to work on projects specifically in preparation for these contests as much as a year in advance—after one year's contest is over, they immediately begin work on an entry for the following year (Bruckman 1997, Bruckman and DeBonte 1997). A new merit-badge system is being designed at the time of this writing to further address the issue of motivation.

```
Storm says, 'monty python is very funny'
Rachael says, 'You can changer your gender by typing "gender me
   as <gender>"'
Rachael says, 'the <>s are where you fill in the blank.'
```

The MOOSE Crossing documentation is written in this style, with angle brackets indicating that you should fill in the blanks. A classic slapstick routine has a bailiff trying to swear in a witness by saying "Repeat after me. I, state your name." The witness replies literally, "I, state your name." Giving someone directions including some things to duplicate literally and others to interpret is awkward, and is a common source of confusion for new users on MOOSE

Crossing. Rachael here is anticipating the problem, giving Storm useful information not just about one command but about how to read the documentation in general.

```
Rachael says, 'Yes, monty python is.'
Storm says, ' i know. I'll love doing that! what planet are you
   from?'
Rachael says, 'Depends.'
Storm says, 'i'm from zork'
Rachael says, 'Of what universe you are referring to-this one,
   star trek...my own..'
Rachael says, 'the game?'
Storm says, 'i'm referring to a computer game.'
Storm says, 'yes. it's fun'
Rachael says, 'I thought so. I've played it.'
Rachael says, 'But I never passed it.'
Rachael says, 'Have you played MYST?'
Storm says, 'what one?'
Rachael says, 'Zapies...I made it up myself when I was in 1st
   grade.'
Storm says, 'no. my mom beat myst.'
Rachael says, 'I haven't yet. It's kinda hard.'
Rachael says, 'Do you live in the Boston area?'
```

Storm here attempted a bit of fantasy role playing. Rachael asks for a clarification—what is the reference frame of the conversation? They move back to talking about real life.

```
Storm says, 'could you show me around?'
Rachael says, 'certainly!@'
Rachael says, 'anywhere in particular?'
Rachael says, 'or just everywhere?'
Storm says, 'everywhere! please'
Rachael says, 'well, let's go up!'
```

When Rachael first offered help, Storm wasn't yet ready to take advantage of it. Now that Storm has tried out a few basic commands and gotten to know Rachael a little bit, she's comfortable asking for help. Rachael leads Storm up to the clouds, where Rachael demonstrates how you can jump and roll around on one of the clouds. While Rachael is jumping around, Storm sets her gender to royal[7], making use of Rachael's earlier offered information. Although Storm and Rachael are not

[7]Changing your gender changes your pronouns. Having the gender "royal" means you are by default referred to by "the royal we."

physically co-located, there is enough of a sense of presence and enough transmission of useful assistance that Rachael helps Storm to progress further within her zone of proximal development (Vygotsky 1978; Newman, Griffin et al. 1989).

Rachael is clearly delighted to meet Storm. She uses this command:

```
→ tell Rally to emote hugs Storm.
```

This causes her pig to hug Storm. Storm hugs the pig back. The pig allows Rachael to express affection indirectly.

Rachael next leads Storm to Home in the Clouds, a room that lets you build your own home by simply typing "build." Storm builds a home there.

Many kids on MOOSE Crossing make use of a special character class that lets them switch between multiple identities. One of Rachael's alter-egos is called QueenAnne. Rachael morphs into QueenAnne and invites Storm to see her castle. Storm types "queenanne" to get to her room without being prompted, and Rachael follows. Storm bows to QueenAnne. QueenAnne changes back into Rachael, and leads Storm to Palladia, part of her mythical kingdom where you can build your own home. Rachael next explains how you can build a home in certain places (not realizing Storm has already done so at Home in the Clouds). Storm builds another home at Palladia. Rachael offers to show Storm her "normal" home, but Storm replies she'd like to do the tutorial. Rachael goes home, returning to her own room in the virtual world. Storm types "tutorial" and then starts looking at the dog tutorials, a three-part sequence that teaches the basics of MOOSE programming by helping you to make your own pet dog. Rachael pages good bye:

```
Rachael pages you.
She pages, 'I'm disconnecting now. I hope you come again.
   Mondays are the best days.'
```

Storm doesn't reply. Rachael disconnects for about a minute, and then logs back on. Storm wanders into Rachael's room and looks at her pet penguin and cat. Rachael is there and says hi. Storm says hi back and goes out a different way than she came in. Rachael follows her. Catching up to her, she says "I can stay somewhat longer." Storm leaves again, and Rachael hesitates and then followed her again, this time to the recycling center. Rachael uses the 'announce' command to print to the room "THE BLUE BIN EATS THE TRASH" followed by "CRUNCH" several times. (Printing unattributed messages like this to everyone in the room is technically against the MOOSE Crossing code of conduct. Members generally remind one another of the code of conduct when it gets out of hand, but tolerate it in moderation if the content stays friendly.) Storm says "I'm getting out of here." She told me during a later interview that she didn't realize Rachael had generated those messages. They walk to town together. At MOOSE Crossing, Storm says "hi! where to." Rachael replies "You choose, I'll follow." They arrive on Main Street:

```
Main Street
You're on the edge of Our Town. Looks like there's space to
   build some shops here!
Obvious exits: ..west.........MOOSE Crossing
               ..north........North Main Street
               ..east.........Town Hall
Storm is here.
Rachael has arrived.
Rally says, 'Hello Clover'
Rally arrives, following Rachael.
Clover arrives, following Rachael.
Rachael says, 'I suggest n'
```
[Rachael is suggesting they continue their explorations by going north. "n" is usually allowed as an abbreviation for north.]
```
Storm says, 'how do you make animals?'
Rachael says, 'Well, it depends if you want a new type of
   animal or one that already exists.'
Storm says, 'new type'
```

Here Rachael checks the command “parents Rally” and gets this response output:

```
Rally(#381)   generic_greeting_creature(#402)   Generic
   Teachable Object (#225)   Generic Puppet(#223)   Generic
   Following Object(#342)   Generic Gendered Object(#77)
   generic thing(#5)   Root Class(#1)


Storm says, ‘i’d like an animal to follow me around’
Rachael says, ‘type “create #223 named <name>”.’
```

Storm does, and chooses the name Jasper. Rachael looks at Storm and sees she’s now holding a creature. She tries to look at it, but Storm is holding it. She asks Storm to drop it. Storm describes it as “a frog” and drops it for Rachael to see.

```
Rachael says, ‘neat idea! I wish I thought of a frog!’
Rachael says, ‘to make it follow you type “set Jasper’s
   following to me’
```

Rachael responds immediately with positive reinforcement, and additional technical advice. Storm improves Jasper’s description to “frog with orange skin, a black stripe and a sense of humor.”

```
Storm says, ‘Jasper likes Star Trek too’
Rachael smiles.
Storm says, ‘how do you make him say things’
Rachael says, ‘okay..now go somewhere, and it will follow you.’
Rachael says, ‘it depends.’
Storm says, ‘ok...’
Rachael says, ‘There are two ways. One, you could make a
   script, so that if you type something it will do something
   in return, like huging Rally..’
Rachael hugs Rally.
Rally squeals happily.
Storm says, ‘how’
Rachael says, ‘if you go to the pencil, it will make a thing
   appear. Fill in the blanks.’
[The pencil icon lets you edit an object, script, or property.]
Rachael says, ‘then, when you have a script ready to fill out,
   you type “on <script’s name> this.’
```

```
Rachael says, 'let me give you an example..'
```
Rachael lists out a script of hers, and says it out loud:

```
Rachael says, 'here:on hug this    tell context "You hug " + my
   name + "."     announce_all_but context context's name + "
   hugs " + my name + "."     emote "squeals happily."   end'
Rachael says, 'where the big blanks are, is where the returns
   are.'
```
Rachael waits a minute, repeatedly checking to see if the script is there yet.

```
Storm says, 'what do you put on the pencil blanks'
Rachael says, 'here, fill in the object as Rally, then fill in
   the script as "hug".'
Rachael says, 'then you'll see what a normal script might look
   like.'
```

Storm first opens an editor on Rally:hug, and then on Jasper:hug. One of the most common learning strategies for kids on MOOSE Crossing is using other kids' programs as examples. In this respect, MOOSE Crossing shares a powerful learning feature with the World Wide Web. As was discussed earlier, on the Web, you can view the HTML code for any object. You don't need to go to a special library of examples—everything is an example. In the course of your normal use of the Web you encounter objects that you can later go back to and use as models. Similarly, every object on MOOSE Crossing is an example. While the MOO language (on top of which MOOSE is built) allows some programs to be unreadable to others, I deliberately chose not to support this feature. All programs can be viewed by everyone and learned from. Children often start with very simple variations on others' programs, and gradually progress to more original creations. This powerful learning strategy is often prohibited in schools, where it is declared to be "cheating." Making work visible makes everyone's thinking explicit, making it easier for both teachers and peers to do formative assessment of work in progress and provide appropriate support.

It's important to note that on MOOSE (unlike the Web), you are likely to know or have the opportunity to meet the creator of each item in the world of examples. Storm is not learning from a randomly selected example; she's learning from an example created by her new friend, Rachael. The social and intellectual relationships are mutually supportive. The available sample programs are situated in a web of social relationships.

```
Rachael says, 'do you understand...or should I try explaining
   some more?'
Storm says, 'wait a sec..'
Rachael says, 'okay..i'll be right back..'
```

Rachael wanders around a bit and comes back to see how Storm is doing. Storm asks how she's supposed to save her work—whether she's supposed to click "Change" or "Revert". Rachael tells her to use "Change".[8]

```
Storm says, 'what do you do after clicking change?'
Rachael says, 'nothing...it should work..'
```

Rachael here lists out Storm's script and sees this:

```
on hug this
tell context "You hug" + "my name+ + "."
announce_all_but context context's name + "hugs" + my name + my
   name + "."     emote "grins widly and jumps with joy."
   end'
Storm says, ' it says I'm missing "end"'
Rachael says, 'hi..sorry...my computer crashed.''
Rachael says, 'you always need to put end at the end.'
```

Storm examines Rally, checks 'parents Rally' and 'parents Jasper' and tries to change the parent object of Jasper to #402, generic greeting creature, a program written by an eleven-year-old boy which serves as the parent object for Rachael's pig. MOOSE is an object-oriented programming environment. Objects "inherit"

capabilities from their parent objects. To change an object's parent, you use the "chparent" command. Storm told me during a later interview that she learned this from the tutorial she read earlier. She gets the syntax slightly wrong (forgetting the word "to") and the chparent doesn't work.

```
Rachael says, 'sorry it crashed again.'
Rachael says, 'it needs to have end on the last line.'
```
Storm reads 'help chparent'.

```
Rachael says, 'like this 1:  on hug this 2:  tell context "You
   hug" + "my name+ + "." 3:  announce_all_but context
   context's name + "hugs" + my name + "."     emote "grins
   widely and jumps with joy." end
Rachael says, 'I mean..'
Rachael says, 'you need to have end on a line below the
   others..'
```

This is somewhat confusing advice. Rachael here checks Storm's last commands. The ability to check someone else's last commands was designed for exactly what Rachael is doing: helping someone figure out what they're doing wrong.

Storm now moves the 'end' to a separate line of her program, and also correctly chparents Jasper to generic_greeting_creature.

```
Rachael says, 'I'm sorry..but I need to go...any questions...?'
Storm says, ' nope! Thanks a lot, Rachael! Bye, Rally, Clover!'
Rachael says, 'bye...see you again soon okay! Can you come by
   monday?'
Rachael says, 'Or this weekend?'
Storm says, 'probably!'
Rachael says, 'By the way, this monday most people won't be
   here...sorry. (because of vacation)'
Rachael says, 'bye!'
Storm says, 'i'll be there, though!'
Rachael says, 'okay..see you then!'
```

---

[8] Based on feedback from users, we've since renamed the button from "change" to "save." Log-file data is useful not only in evaluating educational outcomes, but also in the process of iterative refinement of the software and other elements of the learning environment.

Rachael now goes home, and Storm tries to hug Jasper. She gets this error:

```
OOPS!:  Can't find an object named "my name+ + "
     :  Called from Jasper (#913):hug
     :  Line: <2> tell context "You hug" + "my name+ + "."
```

She gets rid of the stray quote before the "my" in line 2, and next gets an error message from the missing return in line 3. She changes it a few times, and finally gets it to compile, but now gets a run-time error. Next she simplifies the script, eliminating the last few lines:

```
on hug this
  tell context "You hug" + my name + "."
  announce_all_but context context's name + "hugs" + my name +
   "."
```

This gives her a "missing end" error. She now changes it to:

```
on hug this
  tell context "You hug" + my name + "."
  announce_all_but context context's name + "hugs" + my name +
   "."
  emote "grins widely and jumps with joy."
end
```

This compiles, and she tries it out:

```
→ hug Jasper
You hugJasper.
Jasper grins widely and jumps with joy.
```

She edits the program to add in the missing space after the word "hug." Next she looks at herself, adds a few adjectives to her description, and changes her gender from royal to female. She looks at Jasper and hugs him one more time, goes home, and then disconnects for the day.

3. Saturday: Independent Progress

On Saturday, Rachael is on a few times during the day; no one else is around.

Storm logs on in the late afternoon, sees that no one else is on, and wanders around a bit. Next she creates a new creature and calls it Callie. She sets Callie's gender to female and Jasper's to male. She describes Callie as "You see a large, furry cat, with big orange eyes and a royal air around her." Storm is becoming more proficient with the programming environment as well as the language: she uses the client to edit Callie's description property this time, instead of using the "describe" command as she had previously. She adds this script to Callie:

```
on tickle this
  emote purrs
end
```

That works, so now she adds this one:

```
on utter "cat"
  say "I am not a mere cat! I am deeply offended!"
  emote "looks indignant and raises her nose haughtily."
end

on utter "Callie"
  say "It is a beautiful name, is it not? Just like me."
  emote "purrs and sits down in a regal pose."
end
```

MOOSE's pattern-matching parser makes it easy to have multiple scripts with the same name. These new scripts both work as well. When you type "utter cat," everyone in the room sees:

```
Callie says, "I am not a mere cat! I am deeply offended!"
Callie looks indignant and raises her nose haughtily.
```

She looks back at Jasper's hug script again, and next changes Callie's utter scripts to be:

```
on utter "cat"
  tell context "Callie says, 'I am not a mere cat! I am deeply
   offended!'"
  announce_all_but context context's name
  emote "looks indignant and raises her nose haughtily."
end

on utter "Callie"
  tell context "It is a beautiful name, is it not? Just like
   me."
  announce_all_but context context's name
  emote "purrs and sits down in a regal pose."
end
```

In her first version, she was able to put the now familiar "say" and "emote" commands—commands she uses all the time to communicate with Rachael—into her program. Her second version makes her program more like the other program she has seen before. However, she doesn't understand what these commands really do. In this version, when you run the "utter Callie" script:

```
→ utter Callie
[The context (the person typing the command) sees:] It is a
   beautiful name, is it not? Just like me.
[Everyone else but the context sees:] Callie
[Everyone sees:] Callie purrs and sits down in a regal pose.
```

Next she changes the "utter cat" script:

```
on utter "cat"
  tell context "Callie says, 'I am not a mere cat! I am deeply
   offended!'"
  announce_all_but context context's name +  "looks indignant
   and raises her nose haughtily."
end
```

Now the output is:

```
→ utter cat
[Context sees:] Callie says, 'I am not a mere cat! I am deeply

offended!'
```

[Everyone else sees:] `Callielooks indignant and raises her nose haughtily.`

In her first version, the words "Callie says" were generated automatically by the say command. When she switched to "tell" instead of "say," those were missing. She now has manually added them back in. Storm tests this version and observes that she doesn't see Callie look indignant any more, so she puts it back to an 'emote' and makes a similar change to the "utter Callie" script:

```
on utter "Callie"
  tell context "Callie says 'It is a beautiful name, is it not?
   Just like me.'"
  emote "She purrs and sits down in a regal, but dainty, pose."
end
```

This doesn't do quite what she wanted:

```
→ utter Callie
[The context sees:] Callie says 'It is a beautiful name, is it not?
   Just like me.'
[Everyone sees:] From Storm:[9] Callie She purrs and sits down in a
   regal, but dainty, pose.
```

[9]Storm is holding Callie. The system prefaces the output with the words "From Storm" to identify its source.

Next she changes “tell context” in the second script to “tell Callie”. When she tries it, that line simply doesn’t appear to Storm. She puts it back to “context”. In the emote, she changes the “She” to “Callie”, so now she gets a line beginning with “Callie Callie”. Next she changes the start of the line to “emote say.” This prints out “Callie say Callie”. Next she simply scrunches it into one long “tell context” command:

```
on utter “cat”
  tell context “Callie says, ‘I am not a mere cat! I am deeply
   offended!’ Callie looks indignant and raises her nose
   haughtily at you, with a sniff of disgust.”
End

on utter “Callie”
  tell context “She says ‘It is a beautiful name, is it not?
   Just like me.’ She sits in a regal pose, with an air of such
   splendor that makes you feel unworthy of her presence.”
end
```

Now only the person typing the command sees Callie’s response. Storm may not understand that other people won’t see it. In fact, her original version was probably closer to what she wanted. She has spent 29 minutes experimenting. Satisfied, she wanders around the world a bit. She changes Callie’s parent object to make it a greeting creature, as she had done with Jasper. She logs off for the day.

4. Sunday: Social and Technical Support Intertwined

On Sunday, Storm connects again in the morning. Her character wakes up in her home, which she notices she hasn’t yet described. She describes it and then modifies the description a few times, settling on:

```
You are in a black room, with silver stars all over the
ceiling. In one corner, you see a little pond with a fountain
and lots of lily pads. In another, there is a large, comfy
black chair with long, black cat fur all over it. There are two
open windows, and s light breeze gently flaps the black lace
curtains. Silver bells tinkle merrily.
```

Next she reads the dog tutorial parts 1 and 2, and makes a child of generic dog named Toby. She adds an "on tickle this" script to Toby which just "emote laughs". She next changes it to this:

```
on tickle this
   say "That is beneath my dignity."
   emote looks at you with scorn and contempt.
end
```

She looks at the pet script on Toby's parent generic dog, and then uses that as a model to improve her tickle script:

```
on tickle this
  tell context "You tickle " + my name + "."
  announce_all_but context context's name + " tickles " + my
   name + "."
  say "That is beneath my dignity."
  emote looks at you with scorn and contempt.
end
```

The previous day, she struggled to use another object's program as a model. This time, she is successful. This program will tell the context "You tickle Toby" and tell everyone else that "<context's name> tickles Toby." The previous day's experimentation has paid off. It's worth noting also that her program is not an exact copy of what the tutorial or existing dogs do—she has customized it, making her dog react with a different personality. She next tries a couple simple string constructions out at the command line, as suggested by one of the system tutorials (the second part of the dog tutorial):

```
→ "You pet " + my name + "."
=> "You pet Storm."

→ my name
=> "Storm"
```

She next reads help on the dig command, and digs a new room called Cat's Corner, and describes it:

```
You are in a smaller black room, and cat pictures are all over
the place!  Toys and a food dish decorate the floor. A well
used scratching post is in one corner. A big, black sofa with
red cushions is on your right. Lots of cat fur is shed all over
the place. There is a window by the sofa, where bird song is
drifting in.
```

Unfortunately, she realizes that she accidentally put that description on her original room instead of the new one. With a bit of struggling, she manages to switch the descriptions. She told me during a later interview that she did this by reusing different describe commands from her input buffer to switch them.

Next she digs Jasper's Pond off of Callie's Corner. She is careful this time to make sure she is describing the right room. She describes it as:

```
Water splashes into the large pond in the center of this room!
Several frogscroak amid the weeds and stones. You can see some
minnows swim  racefully in the water. Lily pads are abundant
here.
```

Storm wanders around the world a bit and then makes a new object inheriting from generic ticklish object. She told me later that she believes she saw the object number on her visit to the library. She immediately recycles the new object without trying it out. She walks back to the library, looks at the parents of one of the library's objects, goes home, and makes herself another object: catnip mouse. She describes it as " A catnip mouse that seems loved well. Made of red felt and a yarn tail."

At this point, Rachael connects and pages her 'HI!!!' They have both been online several hours since they last met, but not at the same time (see Figure 1). Rachael audits Storm and examines some of her creations remotely. Storm is in the

process of adding a 'throw' script to her catnip mouse. She looks back at the hug script on Jasper as an example. She hasn't responded to Rachael yet, who pages a second time: 'May I join you, oh great power of wind?' Storm pages back 'yes'.

```
Jasper says, 'Welcome Rachael'
Rachael beams in.
Jasper says, 'Welcome Clover'
Clover arrives, following Rachael.
Rachael says, 'hi!'
Jasper says, 'HiRally'
Rally arrives, following Rachael.
Storm says, 'Hi!'
Rachael says, 'What are you doing?'
StormhugsJasper.
Jasper grins widely and jumps with joy.
RachaelhugsJasper.
Jasper grins widely and jumps with joy.
Rachael smiles.
Storm says, 'Just experimenting'
Rachael says, 'ahh...I see you know about dig.'
Storm says, 'yup'
Rachael says, 'Well, you are learning fast!'
Storm says, 'What do you think of my rooms?'
Rachael says, 'They are very nice.'
Storm says, 'Thanks! Did you see Callie'
Storm says, 'she's my cat'
Rachael says, ' could you tell me what :-) means, or what ever
   that thing like it means? I haven't figured it out, and
   everyone does it!'
Rachael says, 'no I haven't. I'd love too.'
Storm says, 'look at it sideways! It's a smily face!'
Rachael says, 'Oh!'
Rachael grins!
```

On Rachael's arrival, Storm immediately shows off her new creation, Jasper's hug script. Rachael acknowledges it by hugging it herself and then smiling. Throughout the conversation, Rachael offers positive feedback. The exchange is nicely two-way: Rachael has a question that Storm can help her with.

Storm next invites Rachael to her other room, where she shows off her cat Callie. Storm suggests that Rachael say "cat." Rachael does, and nothing happens. Storm tries to figure out what's wrong. Rachael looks at the code on the cat and figures out that she needs to type "utter cat", and does. She laughs. This prints output to only her. Storm urges Rachael to try it, not realizing she already

has. Rachael explains she did try it—that's why she laughed. They continue to play with Storm's cat, and then Rachael pets her own cat Clover. Storm pets Clover as well and says she's pretty:

```
Storm says, 'Clover is pretty'
Rachael says, 'I think you are worthy of being a
   Crowned_player_class'
Storm says, 'cool! like what?'
Rachael says, 'Well, you'll see.'
Rachael smiles mysteriously.
Storm says, "wanna see my frog room'
Rachael says, 'sure!'
```

They continue talking, and showing off creations to one another. Rachael explains how her special player class gives you clothes, and suggests Storm use it. Storm explains that her creatures are named after friends of hers—her real life friend Jasper likes frogs. Rachael invites Storm to her game room, which boots you out (sending you back to your home) if you don't type the right thing. She doesn't explain how it works to Storm, but suggests she look at the code to figure it out. After getting booted, Storm just goes idle for a while and doesn't answer Rachael's pages. Rachael goes to work on Clover's enter script, and fixes it. Storm comes back and they talk about the game room a bit. Rachael tells Storm how to use the "list" command to look at code. Storm says she has to go, and they make plans to meet the next day around 3 PM. They both log off.

Rachael comes back a few hours later, and examines some of Storm's things. She makes a small change to one of her own programs, and logs off. Two hours later, she logs on briefly again. Finding no one on, she disconnects. She connects for half an hour again in the evening, and investigates what some basic system objects do.

5. Monday: Emotional Support Stretches the Zone of Proximal Development

Monday (a school holiday), Rachael logs on and off quickly in the morning. Storm connects a little after 1 PM. She starts to work on the dog tutorial, part 3. Rachael notices she's on and pages 'Hi!' Storm pages back 'I'm working on a tutorial. When I'm done, I'll page you again. Thanks!' Rachael pages back 'Okay. See you then!!' Storm works through the tutorial, and then recycles the dog she's made. She goes back to working on the throw script on her catnip mouse. She experiments with variations on using say and emote and leaving them out entirely, with little luck.

Rachael works on improving some of her programs, and then snoops into various system functions. She checks Amy's last commands (which relate to the registration room) and then tries to figure out what the registration room does. Finally, she gets curious about what Storm is doing and checks Storm's last commands. Storm has been tossing her catnip mouse over and over. Rachael asks if she can come over, and Storm says 'sure'.

```
Rachael says, 'hi'
Storm says, 'Hi!'
Rachael says, 'Whatca doin'?'
Storm says, 'Being unsucessful with the dratted mouse!'
Rachael says, 'AIIIE! What seems to be the problem?'
[Rachael here checks 'scripts mouse' and tosses it.]
Rachael says, 'Hmm..what do you want it to do?'
Storm says, 'well, the mouse should be thrown across the room
   but it just won't GO!!!!'
Rachael says, 'okay this is the script:1:  on throw this   2:
   drop   3:    " thrown across room. "    4:  end'
Rachael says, 'well, first, " thrown across room. " dosen't do
   anything. You need to announce it.'
Storm says, ' what the heck is that supposed to mean?'
Rachael says, 'Pardon?'
Rachael says, 'You need to make it say 'announce " is thrown
   across the room."' '
Rachael says, 'I mean, 'announce my name + " is thrown across
   the room."' '
```

Storm changes the script to this:

```
on toss this
  drop
  announce myname + 'is tossed across room.'
end
```

The old throw script is unchanged, which Rachael points out. Storm tells her to try the toss script.

```
Rachael says, 'You made a tiny error. You said 'announce myname
   + 'is tossed across room." when it should say 'announce my
   name + " is tossed across the room."' '
[Storm adds in the space.]
Rachael says, 'When your done could you please drop your dog? I
   want to test something with Clover.'
```

Rachael tries 'toss catnip' and 'catnip mouse is tossed across the room' is printed to both of them.

```
Rachael says, 'okay, now it announces it.'
Rachael says, 'But you want it to go into the room right?'
Storm says, 'yeah'
Rachael says, 'well, I think that if you said 'this:moveto my
   owner's location' it might work..'
Rachael says, 'instead of drop...let's try it.'
Storm says, 'well, let's lay off the mouse before i strangle
   it!!!!!!!!!'
```

Storm tries to take Rachael's advice, but adds the new code to the first line instead of putting it on its own line. Now you can't toss it at all. In my face-to-face discussion with Storm and Rachael together, we were talking about the benefits of working with someone else, and Storm noted particularly appreciating Rachael's support at this moment:

> It was sort of handy when she came in. Like that catnip mouse, which I finally threw away, cause I was just sick of it. And it wasn't working, so.... It was nice to have somebody who I could just sort of rant and rave all my

> woes about this mouse to. [Laughing.] We just started saying "it's ok, it's ok, calm down, it's just a little object, it won't kill you."

In this example, the context of the help is not only significant, it's arguably more important than its content. Storm is in need at this moment of emotional support more than technical support. Not only could that support not be provided effectively by a computer, it also couldn't be provided effectively by very many people. When you are frustrated, you want to talk with a friend. A friend who understands the complexities and frustrations of the problem at hand is particularly valuable. An "intelligent tutoring system" could conceivably have suggested a way to fix Storm's code, but that technical advice is only a small part of the support Storm received from Rachael.

Storm drops her smart dog as Rachael requested, and it reacts to Rachael's cat as Rachael had hoped. Storm compliments Rachael on her cat. Storm suggests they go to the game room, and Rachael agrees. This time Rachael explains how it works right away. After jumping around in the game room for a while, Storm suggests they explore. They go to Paradise Island together, a room built by an eleven-year-old girl where you can swim, climb trees, and build a summer home. Storm builds another home there. Rachael teaches Storm about the 'audit' command. Rachael suggests Storm connect her houses in different places together to make a loop. Storm asks how. Rachael reads "help dig" and gives Storm a concrete example of a command. Storm works on making the circle of exits, with help (and positive reinforcement) from Rachael. Storm runs out of quota, and Rachael advises her to recycle stuff or ask Amy for more quota. (Each member of MOOSE Crossing is given a quota for the total number of objects he or she can build. This is necessary to prevent the database from getting too big for the machine. Children who ask for more quota are almost always granted more.)

Rachael has to leave. They plan to meet the next day at 10:30 AM. Rachael logs off and Storm logs off. Rachael logs on again a few minutes later. I am back from Maine, and talk to her a while later:

```
Rachael laughs. "Have you seen any of Storm's things
Amy says, 'she's made stuff already?'
Amy says, 'wow!'
Rachael says, 'Yes! Will almost full credit!'
Amy says, 'she's been busy!'
Rachael says, 'Yep! I've showen her somethings, but she's done
   some on her own.'
Rachael says, 'It is. She is really nice.'
Rachael says, 'We spent almost 2 hours together today.'
Amy says, 'great!'
```

Rachael is logged on for several more hours that day. She programs Marj, a robot that acts as a second self for her online.

6. Tuesday: Collaboration and Role Evolution

Storm logs on early in the morning. She continues to work on the dog tutorial, part 3. As part of her work on the tutorial, she has made a pet dog named Tao and another pet called Melvin the Moose. She changes the parent of her objects from generic dog to generic cliché-spouting object, a program I wrote that inserts clichés into the conversation at random intervals, and to which you can teach new clichés. She disconnects.

Rachael and Storm both connect promptly at 10:30, the time at which they had agreed to meet. Rachael sends her new puppet, Marj, to talk to Storm. Storm teaches Melvin a cliché. Rachael drops Clover. She's made some coding improvements on Clover since they last talked. Each of them is immediately showing off the new work they've done.

Storm must be a bit confused, because she pages Rachael asking if she would like to join her. Rachael has Marj explain that she is already there. As Rachael is talking, Melvin the Moose is generating errors.

```
Marj looks distasefully at Melvin. It is giving Rachael lots of
   errors because she is talking through a puppet.
Marj says, 'I am not a MEAR puppet. I am a non - orgainic
   lifeform.'
Storm says, 'I am sorry about my moose. She is very...strange.'
```
[What Storm is saying isn't actually getting across to Rachael, because Rachael's puppet isn't working correctly yet.]
```
Marj says, 'We non - orgainic lifeforms have feelings too!'
(from Rachael's room) Rachael grins broadly.
Storm says, 'And I know, o mighty one, that you are not a MERE
   puppet!'
```
[Rachael again doesn't hear this line.]

The conversation continues, with Rachael not hearing Storm's side. Rachael wonders why Storm is being so quiet. Storm responds that she's not; Rachael of course doesn't hear that either. Storm asks a question, and gets no answer. She repeats it with increased urgency (i.e. six question marks). Finally, Rachael checks Storm's last commands and realizes something is wrong. In this instance, Storm has helped Rachael to detect a bug in Rachael's program. The exchange of technical help has already become very much two way—the girls have a reciprocal interdependence (Cohen 1994).

Storm suggests they make a medieval room. Rachael likes the idea. Storm asks where they should make it. Rachael suggests off of her castle. Rachael makes the room. Storm asks if she can be the one to describe it, and Rachael says yes. (Joint ownership of objects is not currently supported. It would be a desirable feature.) In this case, Storm ends up building an additional room so that she can be the one to describe it. Rachael sets up the exits to connect the new room to her castle. Storm writes a first description, and Rachael offers suggestions for additions. Storm starts with:

```
You are in a red room. Armor and weapons are all over, but they
are old and dull, so you can't use them. But the armor and
weapons are strikingly well made.
```

This conversation ensues:

```
Storm says, 'hi. Look here and tell me what you think...'
```
[Rachael isn't in the room at this moment, and doesn't hear this line.]
[Rachael arrives.]
```
Rachael says, 'hi'
Rachael says, 'hmm...'
Rachael says, 'Great!'
Rachael says, 'The only problem is the exits...look it says
   passage...armory'
Rachael says, 'We can have amy help us fix that.'
Storm says, 'any other ideas for describing?'
Rachael says, 'wooden benches?'
Rachael says, 'blood stains?'
Rachael says, 'rusty weapons?'
Storm says, 'i like it! give me a second...'
Rachael says, 'a fire'
Rachael says, 'ok'
Rachael says, 'it is dim'
Rachael says, 'the only light is from the window and fire'
Rachael says, 'the fire is roaring'
Rachael says, 'in the center of the room.
Rachael says, 'We could rename the room to "hero's hall!"'
Rachael says, 'I mean Hero's Hall'
Storm says, 'try the description now...'
```

Rachael's brainstorming inspires Storm to improve the description:

```
You are in a red room. Armor and weapons are all over, but they
are old and dull, so you can't use them. But the armor and
weapons are strikingly well made. Tapastries of unicorns and
knights decorate the walls. There is a lone window here, with
red velvet curtains. You see a blood stain, old and brown,
streaked across the wall. The only light comes from the fire,
for in this room it is eternally night. The fire is in the
center of the room, and it is crackling wickedly. You see a
white flutter in a shadowed corner, and your heart stands
still. But it is only a piece of cloth...or so you think...
Rachael says, 'wow! It's great!'
Storm says, 'Thanks.'
Storm says, 'I love writing, you see...'
Rachael says, 'It's like a murder mysteroy'
```

We discussed the construction of the room in our face-to-face interview:

Amy: Which parts of that were whose idea? How did it start, and whose idea was which part?
Rachael: I think it was her idea originally to build the room
Storm: Yeah, I wanted to build a room that we could both do stuff in. And she suggested a castle. And castles and dungeons and dragons are just stuff I've always liked. When I was like two I had a dream about dragons that would take me places. So I always liked them and I thought, that would be pretty cool if we did all this stuff. I wanted to describe it cause I like to write. My original description was like one sentence, which was boring.
Amy: And then what happened?
Storm: Then she [turning to Rachael] then you said all the suggestions, you know, like blood stains and stuff like that. And I thought "Ah yeah!" And I thought of a catalog that we get, it's called Design Toscano. It's got gargoyles and tapestries, swords and that kind of stuff. So I just added a whole bunch of stuff.
Rachael: And I think after I saw her description I thought we should add some objects or something in the room or something to follow up the description.
Storm: Like blood stains. And I thought "Ah ha! Yes, There we go! Now I have some ideas"
Amy: Do you think either of you could have done as good a job on your own?
Storm: No.
Rachael: No, I don't think so.
Amy: Is it fun working with someone else?
Rachael: It's much more fun than working by yourself. Cause if there's nobody to see it, why even bother doing it?

This is a nice example of collaboration. The girls concur that the result is better than if either girl had made it on her own. They are learning from each other and inspiring each other to do better work. They have given each other constructive feedback, and each served as an appreciative audience for the other's creative efforts. Storm's role has matured from tutee to design partner. This natural role evolution is an essential element of apprenticeship and cognitive apprenticeship learning models (Collins, Brown et al. 1989; Lave and Wegner 1991). The girls both describe this as a very positive experience.

In reviewing the literature on collaboration, Elizabeth Cohen notes that ill-structured problems are particularly well suited to collaborative learning. Creative writing is a classic open-ended problem where collaboration can be beneficial. Cohen writes that "given a problem with no one right answer and a learning task that will require all students to exchange resources, achievement gains will depend

on the frequency of task-related interaction" (Cohen 1994). The highly social nature of the MOOSE environment encourages such interaction. It should be noted that while Cohen emphasizes task-related interaction, task-independent interaction forms social bonds which support task-related interaction.

Rachael and Storm decide the room needs a dead body and note. Is it a murder or a suicide? Who is dead and why? Their shared vision of what has happened here evolves as they work together. This time Rachael writes, and Storm offers detailed feedback. Rachael says she has to go. They agree to meet Thursday at 5.

Rachael comes back half an hour later. She asks Storm if she's going to go the Media Lab. Storm doesn't understand; Rachael explains that some times people go to the Media Lab to do MOOSE Crossing. Storm says no, she's not going. Storm shows Rachael her new Gargoyle. It has a couple of errors: it's an adaptation of the smart dog tutorial, but it's missing some required properties. Rachael explains that to Storm, and Storm fixes it.

Storm has made a ghost object. Its description reads:

```
The ghost of the murdered girl. She says "I will not harm you.
I am left here to mourn my loved one forever. Heaven would not
take me because of my suicide, but Hell could not take me
because of my goodness. So I must stay here, and mourn Patrick
until the stars grow cold and the Earth is no more."
```

Rachael suggests Storm might make the ghost say that whenever anyone enters the room. She explains how to write an 'enter' script.

Earlier that day, I met and talked with Storm online. She showed me her creations, and I complimented her on them and on her fast progress in general. Storm mentions this encounter to Rachael. Despite the child-centered nature of the environment, feedback from adults is still significant:

```
Storm says, 'oh, amy loved my work and gave me some more quota
   to make stuff..'
Rachael says, 'That's great!'
```

```
Rachael says, 'she did the same to me, byron, zoro, and
   miranda.'
Storm says, 'I'm going to try to make that script... Why don't
   you create a sword or something? Make it out of Generic
   Jewelry, if there is such a thing...'
```

Rachael and Storm hang out together in the same room. Rachael works on getting Marj to be able to page people; Storm writes the enter script Rachael suggested. The girls' online friendship continued for more than a year, much in the same fashion.

7. Seven Months Later: Meeting Face to Face

The girls finally met face to face for the first time on the day that they both came to the Media Lab to talk with me about this article. They hadn't met previously because they live more than a 90-minute drive apart, and because the MOOSE Crossing code of conduct strongly discourages meeting other members face to face for safety reasons. Storm said Rachael was much like she had imagined. Rachael confessed that she hadn't really given much thought to what Storm was like in real life. Storm arrived at the Media Lab first, and I interviewed her separately. I had interviewed Rachael on several previous occasions. Next Rachael arrived, and I interviewed the two of them together. I then gave them both an hour to talk without me present, and went to chat with their parents. The girls immediately decided to log on to MOOSE Crossing. Afterwards, I asked them to compare connecting together versus connecting alone from home:

> Amy: How is doing this sitting next to each other at two terminals different from doing it from two towns a hundred miles away?
> Storm: It can be a lot more funny, cause we can talk to each other.
> Rachael: Yeah, like when they're at the Media Lab, at least in the beginning especially, they'd be talking and they'd say like "that's not funny." And I'm like, what are they talking about? And then when I went to the

Media Lab,[10] I realized they're joking around at the same time. So we were just doing that to Zeus. I told him eventually what was going on.
Amy: So it's more fun with someone in the same room?
Storm: It's a lot of fun that way.
Amy: Do you think it would be just as much fun if everyone was always in the same room?
Rachael: Well, no.
Storm: That won't work if everyone was just in the same room.
Amy: Well, it *could* be. We could just have the people at the Media Lab and that's it.
Rachael: It's fun to have people you've never seen before. And it's fun, the aspect of it being global.

Rachael and Storm immediately liked each other face to face. After leaving the Media Lab late that Sunday afternoon, their families went out to dinner together. A few days later, I asked them each via MOOmail (email internal to MOOSE Crossing) how she thought meeting face to face would affect their online friendship. Rachael responded "I'm not sure. I think that it might strengthen our friendship...but I honestly don't know." Storm said "I don't think it really changed our friendship much at all." They hope to meet up face to face again, but the driving distance between them may be an obstacle.

Rachael's experience meeting Storm stands in contrast to her experience meeting another MOOSE-Crossing member named Miranda (girl, age 11). Rachael describes herself as "unpopular" and says "I'm a very intellectual person. I'm not very physical." I would characterize Storm as earthy. On meeting face to face, they appeared immediately comfortable with one another. On the other hand, I would characterize Miranda as a popular child. (Miranda participated in a MOOSE Crossing after-school program, and was observed and interviewed regularly over an eighteen-month period as part of a prior study (Bruckman 1997). Miranda's mother was also interviewed.) Miranda is smart, pretty, and adept and relaxed in her social relations. Rachael and Miranda had developed a warm friendship online. However, on meeting face to face, they seemed immediately uncomfortable with one

[10]Rachael is referring to the children who come to the regular MOOSE Crossing after school program at the Media Lab. She attended that program occasionally—three times in 1996.

another. Online, they hadn't been aware that one of them was a popular sort of kid, and the other was unpopular. Rachael says that after meeting Miranda face to face, she was initially much less comfortable with her; however, over the months that followed, she slowly became more comfortable again. Different factors are significant in determining social success and social compatibility online versus face to face.

8 Conclusion: Integrating Technological and Social Contexts

Storm's experiences that weekend are quite typical of the general patterns of interaction on MOOSE Crossing. However, she was unusually lucky in one key respect: she immediately met someone with whom she has significant shared interests. At the time of this writing, the population of MOOSE Crossing is still smaller than is desirable. This makes it more difficult for individuals to meet people they particularly like, because there are fewer people to chose from. Even within a larger community, it may take time before a new member happens to meet someone with shared interests. Storm was fortunate to meet Rachael right away.

Support structures for learning are available to the learner on MOOSE Crossing, but not imposed on them. This is consistent with Elizabeth Cohen's conclusion that "when the teaching objective is learning for understanding and involves higher order thinking, task arrangements and instructions that constrain and routinize interaction will be less productive than arrangements and instructions that foster maximum interaction, mutual exchange, and elaborated discussion" (Cohen 1994). MOOSE Crossing is designed to maximize opportunities for exchange.

Of particular interest is the inseparability of the social and intellectual activity going on. It matters not just that a learner have access to adequate support. One must also ask, support from whom or what? Byron Reeves and Clifford Nass have demonstrated that people often relate to computers using the strategies and skills

they learn from relating to people (Reeves 1996). What is the relationship between the learner and the person or program offering assistance? That relationship is rarely if ever neutral. If a computer program attempts to affect a disembodied, non-anthropomorphized tone, working with that program will still evoke the learner's basic feelings about computers and technology in general. For many people, those feelings are not positive. For this reason, the tutorials on MOOSE Crossing are written in a first-person, chatty style. The dog tutorial begins, "Hi there! This is Amy and I thought I'd show you how I made my dog, Pumpernickel." (Storm noticed the tone of the tutorials, and commented that it was just right—"It's not totally perky, but it's not like [speaking in a monotone] 'this is how you do it.'") The informal, warm tone is intended to put kids at ease.

While some attention to the "personality" of a computer program can help make a more supportive learning environment, it's not clear to what extent that can ever substitute for human contact. A human can, for example, tailor assistance to the particular situation. Rachael was responding to Storm in a way computer programs can not, at least yet. Newman et al. comment that "current computer systems are actually quite far from being able to perform the feats of sensitive interpretation performed routinely by human teachers" (Newman, Griffin et al. 1989). Research in artificial intelligence and education aims to help tailor support to the user, often by attempting to model what the user knows and anticipating typical mistakes. One might argue that it will be necessary to solve the more general problem of creating an artificial intelligence before such tutoring systems can be truly successful. Regardless of whether that is true, the key point here is that providing support for learning is not just a matter of delivering the right content—the context is also important.

Even if a hypothetical computer program could provide the exact same content as a human teacher, the learning outcome would not necessarily be the same, but

would depend on the student's feelings about the tutor (whether human or computational.) Herbert Kohl's work illustrates how powerfully such feelings can influence learning outcomes. In his book *I Won't Learn from You*, Kohl documents how students often may not be *failing* to learn, but instead may be *choosing* not to learn because they do not wish to be subjugated to the authority of the teacher (Kohl 1994). At issue is not the content of what is being said in the classroom as much as who is saying it.

Similarly, in *Jocks and Burnouts: Social Categories and Identity in the High School,* Penelope Eckert tells a compelling story of why some students may choose to embrace the goals set for them by adult authority, and others may reject those goals. While much research in the Piagetian tradition tends to assume that all subjugation of children to adult authority is undesirable (Falbel 1989), Eckert's sociological work is more balanced: she notes that while the "burnouts" reject adult authority, the "jocks" thrive under it. She makes a compelling case that social class (of both students and teachers) is a major determiner of which children chose to embrace or reject goals set for them by authority figures. The picture to be painted here is a complex one, embracing many factors beyond the simple issue of the content of instruction.

In addition to who provides help, there is also the question of *where* it is provided. Where children learn can be as important as what they learn. Where support for learning is provided is another dimension of its context. Kohl and Eckert document how many children reject the entire context of school. Places like MOOSE Crossing and The Computer Clubhouse strive to create new learning cultures with more positive connotations. The Computer Clubhouse is an open drop-in center associated with The Computer Museum in Boston where children can work on creative projects using technology (Resnick, Rusk, et al. 1996). Learning there is self-motivated, and the atmosphere is conducive to collaboration. Many

children who have rejected traditional schooling have embraced the learning culture at The Computer Clubhouse (Resnick, Rusk et al. 1998). While some radical theorists envision a future without schools in the traditional sense (Papert 1980), a more moderate perspective asks whether any of the positive qualities of places like The Computer Clubhouse and MOOSE Crossing can be appropriated in the process of school reform.

Storm received help not from just anyone, but from a girl her own age who also loves *Star Trek* and *Monty Python*. Similarly, she used as examples not some official library of sample programs written by a disembodied authority figure, but programs designed and actively used by her new friend, Rachael. She also looked at additional projects that Rachael recommended, written by Rachael's other online friends. Her choice of examples to embrace is a significant act that takes place in the context of a network of social relations. A child who mimics another's way of dressing or speaking is paying that child a high compliment, and taking on that child as a role model. Similarly, using someone's project as an example to learn from is to some extent accepting the project's author as a role model. Making use of a sample program can be as much a social as an intellectual act.

Social context is of central importance to any learning experience. One of the strengths of networked learning environments is their ability to help integrate a supportive social context with the computational context. The Logo language traditionally arrives in a shrink-wrapped box, with little social support for its use. Teacher workshops try to build networks of support for the use of Logo as a learning tool; however, the workshops are usually too short, and too few teachers can attend. Once a workshop is over, typically so is the support. Additionally, the model most often used is one to many: a workshop leader supports a group of teachers; each teacher supports many kids. In a many-to-many model, kids support one another. MOOSE Crossing attempts to deliver a rich computational artifact to

kids, with a network of many-to-many social support built in. The constructionist model that motivated the design of Logo is inspiring, but the reality of its use in real settings often falls short (O'Shea 1997). This research explores how the Internet can help to provide social support to make a constructionist environment work in realistic settings.

Of central importance to any learning environment is whether the students are motivated to learn. In the concluding chapter of *CSCL: Theory and Practice of an Emerging Paradigm*, Kolodner and Guzdial pose a series of questions of interest to the future of research on Computer-Supported Collaborative Learning (CSCL). Among those questions, they ask "How do we structure CSCL activities to encourage the kinds of motivation that lead to better learning?" (Kolodner and Guzdial 1996). Storm and Rachael were highly motivated to learn over the course of this weekend—so motivated that they chose this activity over all the possible things children could be doing on a holiday. A number of factors contributed to Storm and Rachael's motivation to learn over this weekend. The CSCL environment they used is:

- Richly social,
- Connected to popular culture,
- Open-ended, and
- Rewarding of creativity and individuality.

Children's natural interests in popular culture and in socializing with one another are often suppressed by adults. Children are scolded to stop reading comic books, stop watching television, stop talking to one another, and get down to the business of learning. What if they approached their learning with the same enthusiasm that they approach pop culture and social activity? Instead of working to suppress those activities, educators can leverage off of children's natural interests to foster a genuine interest in learning.

Some critics argue that learning should NOT be fun—we need to teach children the value of hard work and discipline, and not pander to their whims. As a culture, we are in danger of "amusing ourselves to death" (Postman 1985). There are elements of truth in this argument—certainly children can't be expected to be entertained all the time. However, hard work and serious learning are not inconsistent with fun. We can strive to make learning, in Seymour Papert's words, "hard fun" (Papert 1996). The social nature of CSCL environments makes them particularly well suited to realizing this vision.

8.1 Design Recommendations

In a 1996 *Interactions* article entitled "Pianos, Not Stereos: Creating Computational Construction Kits," Mitchel Resnick, Fred Martin, and I noted the importance in the design of learning environments of creating *connections*:

> 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.
>
> • Epistemological connections. Construction kits and activities should connect to important domains of knowledge—more significantly, encourage new ways of thinking (and even new ways of thinking about thinking). A well-designed construction 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. (Resnick, Bruckman, et al. 1996)

These design ideas are useful not just for construction kits, but for many kinds of learning environments. Implicit but not given sufficient emphasis in those

guidelines is the importance of support for learning. Connections are especially important in the design of learning support. Extending this framework accordingly, this paper adds a third design guideline: support connections (or "situated support"):

> • Situated support. Support for learning should be from a source (either human or computational) with whom the learner has a positive personal relationship, ubiquitously available, richly connected to other sources of support, and richly connected to every-day activities.

Acknowledgements

I'd like to thank the reviewers for their thoughtful comments. Joshua Berman, Priscilla Dodds, Mark Guzdial, and Jim Hudson read and gave helpful feedback on drafts of this paper. Joshua Berman and Priscilla Dodds provided invaluable assistance with background research. Thanks also to Austina DeBonte, Elizabeth Edwards, and Jason Ellis. Mitchel Resnick and Henry Jenkins helped shape an earlier version of this paper. Special thanks to Storm and Rachael, and to their parents.

Research in the Electronic Learning Communities (ELC) group at Georgia Tech is supported by grants from IBM, Intel, Microsoft, The National Science Foundation, Neometron, and Ricoh. For more information about the ELC group, see: http://www.cc.gatech.edu/elc/

Bibliography

Anderson, J., A. Corbett, et al. (1995). "Cognitive Tutors: Lessons Learned." *The Journal of the Learning Sciences* **4**(2): 167-207.

Aspnes, J. (1992). Personal email communication.

Bartle, R. (1990). "Interactive Multi-User Computer Games." MUSE Ltd. Available as ftp://ftp.lambda.moo.mud.org/pub/MOO/papers/mudreport.txt

Bruckman, A. (1997). "MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids." Ph.D. dissertation, MIT.

Bruckman, A (1998). "Community Support for Constructionist Learning." Computer Supported Cooperative Work 7: 47-86.

Bruckman, A. and A. DeBonte (1997). "MOOSE Goes to School: A Comparison of Three Classrooms Using a CSCL Environment." Proceedings of CSCL, Toronto, Canada.

Cherny, L. (1995). "The MUD Register: Conversational Modes of Action in a Text-Based Virtual Reality." Ph.D. dissertation, Stanford University.

Cherny, L. (1999). *Conversation and Community, Chat in a Virtual World*. Stanford, CA, CSLI Publications.

Cohen, E. G. (1994). “Restructuring the Classroom: Conditions for Productive Small Groups.” *Review of Educational Research* **64**(1): 1-35.

Collins, A., J. S. Brown, et al. (1989). "Cognitive Apprenticeship: Teaching the Crafts of Reading, Writing, and Mathematics." *Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser* Ed. L. B. Resnick. Hillsdale, NJ, Lawrence Erlbaum Associates. 453-494.

Davis, E. (1998). "Scaffolding Students' Reflection for Science Learning." Education. Berkeley, CA, University of California.

Falbel, A. (1989). "Friskolen 70: An Ethnographically Informed Inquiry Into the Social Context of Learning." Ph.D. dissertation, MIT.

Geertz, C. (1973). *The Interpretation of Cultures*. New York, Basic Books.

Guzdial, M. (1994). "Software-Realized Scaffolding to Facilitate Programming for Science Learning." *Interactive Learning Environments* **4**(1): 1-44.

Jackson, S. L., J. Krajcik, et al. (1998). "The Design of Guided Learner-Adaptable Scaffolding in Interactive Learning Environments." CHI, Lost Angeles, CA, ACM.

Jenkins, H. (1992). *Textual Poachers, Television Fans and Participatory Culture*. New York, Routledge.

Johnson, D. W. and R. T. Johnson (1987). *Learning Together & Alone*. Englewood Cliffs, NJ, Prentice-Hall.

Kohl, H. (1994). *I Won't Learn from You*. New York, The New Press.

Kolodner, J. and M. Guzdial (1996). "Effects With and Of CSCL: Tracking Learning in a New Paradigm." *CSCL: Theory and Practice* Ed. T. Koschmann. Mahwah, NJ, Lawrence Erlbaum Associates.

Koschmann, T., Ed. (1996). *CSCL: Theory and Practice*. Mahwah, NJ, Lawrence Erlbaum Associates.

Lave, J. and E. Wegner (1991). *Situated Learning : Legitimate Peripheral Participation*. Cambridge, England, Cambridge University Press.

Mankins, D. (1999). Personal email communication.

Negretti, R (1999). "Web-Based Activities and SLA: A Conversation Analysis Research Approach." *Language Learning & Technology* 3:1, 75-87.

Newman, D., P. Griffin, et al. (1989). *The Construction Zone: Working for Cognitive Change in School*. Cambridge, England, Cambridge University Press.

O'Shea, Tim (1997). "Mindstorms 2." *Journal of the Learning Sciences* 6:4, 101-108.

Papert, S. (1980). Mindstorms: Children, Computers, and Powerful Ideas. New York, Basic Books.

Papert, S. (1985). "Love Is a Better Master than Duty." Proposal to the National Science Foundation.

Papert, S. (1996). *The Connected Family*. Marietta, GA, Longstreet Press.

Perkins, D. (1996). "Teaching for Understanding." AERA, New York.

Piaget, J. (1929). *The Child's Conception of the World*. New York, Harcourt, Brace and Company.

Postman, N. (1985). *Amusing Ourselves to Death, Public Discourse in the Age of Show Business*. New York, Viking.

Reeves, B. and C. Nass (1996). *The Media Equation : How People Treat Computers, Television, and New Media Like Real People and Places*. New York, Cambridge University Press.

Resnick, M., A. Bruckman, et al. (1996). "Pianos, Not Stereos: Creating Computational Construction Kits. *Interactions* 3:6.

Resnick, M. and N. Rusk (1996). "The Computer Clubhouse: Preparing for Life in a Digital World." IBM Systems Journal 35(3-4): 431-440.

Resnick, M., N. Rusk, et al. (1998). "The Computer Clubhouse: Technological Fluency in the Inner City." In *High Technology and Low-Income Communities*. Edited by D. Schon, B. Sanyal, and W. Mitchell. Cambridge, MA: MIT Press.

Rheingold, H. (1993). *The Virtual Community: Homesteading on the Electronic Frontier*. Reading, MA, Addison-Wesley Publishing Company.

Rogoff, B. (1994). "Developing Understanding of the Idea of Communities of Learners." *Mind, Culture, and Activity* **1**(4): 209-229.

Roschelle, J. (1996). "Learning by Collaborating: Convergent Conceptual Change." *CSCL: Theory and Practice of an Emerging Paradigm* Ed. T. Koschmann. Mahwah, NJ, Lawrence Erlbaum Associates.

Schön, D. A. (1987). *Educating the Reflective Practitioner*. San Francisco, Jossey-Bass.

Seidman, I. (1998). *Interviewing as Qualitative Research*. New York, Teachers College Press.

Spitulnik, M. W. (1999). "Teaching for Understanding: One Teacher's Use of Instructional Strategies to Support Model Building." National Research in Science Teaching meeting, Boston, MA,

Vygotsky, L. (1978). *Mind in Society*. Cambridge, MA, Harvard University Press.