40

HCI FOR KIDS

Amy Bruckman, Alisa Bandlow, and Andrea Forte
Georgia Institute of Technology

DESIGNING FOR AND WITH CHILDREN

How is designing computer software and hardware for kids different from designing for adults? Many researchers have addressed questions about the impact of technology on children; less has been said about the impact children can have on the design of technology. Methods for designing with and for children are only recently becoming widespread features of the design literature (see Jensen & Skov, 2005).

In designing for children, people tend to assume that kids are creative, intelligent, and capable of great things if they are given good tools and support. If children cannot or do not care to use technologies we have designed, it is our failure as designers. These assumptions are constructive, because users generally rise to designers' expectations. In fact, the same assumptions are useful in designing for adults. Designers of software for children start at an advantage because they tend to believe in their users. However, they may also be at a disadvantage because they no longer remember the physical and cognitive differences of being a child.

In this chapter, we will

- Describe how children's abilities change with age, as it relates to HCI
- Discuss how children differ from adults cognitively and physically, for those characteristics most relevant for HCI
- Discuss children as participants in the design process
- Review recommendations for usability testing with kids
- Review genres of computer technology for kids, and design recommendations for each genre

HOW ARE CHILDREN DIFFERENT?

As people develop from infants to adults, their physical and cognitive abilities increase over time (Kail, 1991; Miller, L. T. & Vernon, 1997; Thomas, 1980). The Swiss psychologist Jean Piaget was a leading figure in analyzing how children's cognitions evolve (Piaget, 1970). Piaget showed that children do not just lack knowledge and experience but also fundamentally experience and understand the world differently than adults. He divided children's development into a series of stages:

- Sensorimotor (birth to 2 years)
- Preoperational (ages 2 to 7)
- Concrete Operational (ages 7 to 11)
- Formal Operational (ages 11 and up; Piaget, 1970, pp. 29–33)

Contemporary research recognizes that all children develop differently, and individuals may differ substantially from this typical picture (Schneider, 1996). However, this general characterization remains useful.

In the sensorimotor stage, children's cognitions are heavily dependent on what their senses immediately perceive. Software for children this young is difficult to design. Little interaction can be expected from the child. Obviously, all instructions must be given in audio, video, or animation, since babies cannot read. Furthermore, babies generally cannot be expected to use standard input devices like a mouse effectively, even with large targets.

"Reader Rabbit Toddler" (www.thelearningcompany.com) is targeted at children ages one to three. To eliminate the need for mouse clicking, the cursor is transformed into a big yellow star with room for five small stars inside it. As the mouse is held over a target, the small stars appear one at a time. When the fifth star appears, it counts as clicking on that target. If the child does click, the process simply moves faster. The only downside is the occasional unintended click on the "Go back to the main menu" icon.

In most activities in "Reader Rabbit Toddler," nearly random mouse movement will successfully complete the activity. For example, in the "Bubble Castle" activity, the child needs to rescue animals trapped in soap bubbles that are bouncing around the screen. Random mouse movements will catch the animals relatively quickly. Yet, the parent or teacher watching a child's use of the software over time will typically begin to detect patterns in that mouse movement that become more obviously intentional—the mouse moves more directly toward the bubbles with animals in them. This is a particularly well thought out interface because it mimics how young children learn language. A baby's first attempts at sounds are greeted with great enthusiasm—the child says an unrecognizable phoneme and the parents smile and say, "You said Dada! This is Dada!" Over time, the utterance begins to really sound like the child said "Dada." An initial positive reinforcement for even the most remote attempt at the target behavior puts the child on a good learning trajectory to acquiring that behavior (Holdaway, 1979).

Many examples of software and other cultural artifacts for young children are designed in accordance with adult expectations of what a child should like. There are a few noteworthy exceptions—for example, the television show Teletubbies is out of harmony with those stereotypes. Many adults find the television show bizarre and grating, but it is wildly popular with toddlers. The designers of the original BBC television series, Davenport and Wood (1997), used detailed observations of young children's play and speech in their design. Wood commented:

> Our ideas always come from children. If you make something for children, the first question you must ask yourself is, "What does the world look like to children?" Their perception of the world is very different to that of grown-ups. We spend a lot of time watching very young children: how they play; how they react to the world around them; what they say. (Davenport & Wood, 1997)

Focus groups also played an important role (BBC, 1997). Young children are so radically different from adults that innovative design requires careful fieldwork.

While toddlers' interactions with software on a standard desktop computer afford limited possibilities, specialized hardware can expand the richness and complexity of interactions. For example, "Music Blocks" (www.neurosmith.com) is recommended for ages two and up. Five blocks fit in slots in the top of a device rather like a "boom box" portable music player. Each block represents a phrase of music. Each side of the block is a different instrumentation of that musical phrase. Rearranging the blocks changes the music. Interaction of this complexity would be impossible for two-year-olds using a screen-based interface, but is quite easy with specialized hardware. Research on alternative computer interfaces such as tangible technologies (Ishii & Ullmer, 1997; Dourish, 2001) holds great promise for novel children's interface designs (Price, Rogers, Scaife, Stanton, & Neale, 2003; O'Malley & Stanton, 2004).

In the preoperational stage (ages 2 to 7), children's attention spans are brief. They can hold only one thing in memory at a time. They have difficulty with abstractions. They cannot understand situations from other people's points of view. While some children may begin to read at a young age, designs for this age group generally assume the children are still preliterate. It is reasonable to expect children at this age can click on specific mouse targets, but they must be relatively large. Most designers generally still avoid use of the keyboard (except "hit any key" approaches).

In the concrete operational stage (ages 7 to 11), "We see children maturing on the brink of adult cognitive abilities. Though they cannot formulate hypothesis, and though abstract concepts such as ranges of numbers are often still difficult, they are able to group like items and categorize" (Schneider, 1996, p. 69). Concrete operational children are old enough to use relatively sophisticated software, but still young enough to appreciate a playful approach. It is reasonable to expect simple keyboard use. Children's abilities to learn to type grow throughout this age group. It is reasonable to expect relatively fine control of the mouse.

Finally, by the time a child reaches the formal operational stage (ages 11 and up), designers can assume the child's thinking is generally similar to that of adults. Their interests and tastes, of course, remain different. Designing for this age group is much less challenging, because adult designers can at least partially rely on their own intuitions.

Using age as a guide can be useful; however, designers should be aware that designing too young can be just as problematic as designing too old. Children are acutely aware of their own abilities; being asked to interact with technology designed for younger children can be perceived as an affront or boring (Halgren, Fernandes, & Thomas, 1995; Gilutz & Nielsen, 2002).

FIGURE 40.1. Children playing with Music Blocks.

In addition to considering the implications of cognitive development on children's abilities to use technologies, it is important to remember that different age groups differ culturally too. Understanding what is fun or interesting for a particular age group involves understanding both children's developmental abilities and children's culturally-dependent aesthetic sensibilities. Oosterholt, Kusano, and Vries (1996) suggested that designers should avoid trying to be fashionable—what is cool changes quickly—and should target a limited age range because children's abilities and sensibilities change quickly as well.

In the next sections, we will focus on several characteristics of children that are relevant for HCI research:

- Dexterity
- Speech
- Reading
- Background knowledge
- Interaction style

Dexterity

Young children's fine motor control is not equal to that of adults (Thomas, 1980), and they are physically smaller. Devices designed for adults may be difficult for children to use. Joiner, Messer, Light, and Littleton (1998) noted that "the limited amount of research on children has mainly assessed the performance of children at different ages and with different input devices" (p. 514).

Numerous studies confirm that children's performances with mice and other input devices increase with age (Joiner et al., 1998; Hourcade, 2002). Compared to adults, children have difficulty holding down the mouse button for extended periods and have difficulty performing a dragging motion (Strommen, 1994). This means that many standard desktop interface features pose problems for young users. For example, kids have difficulty with marquee selection. Marquee selection is a technique for selecting several objects at once using a dynamic selection shape. In traditional marquee selection, the first click on the screen is the initial, static corner of the selection shape (typically a rectangle). Dragging the mouse controls the diagonally opposite corner of the shape, allowing you to change the dimensions of the selected area to encapsulate the necessary objects. Dragging the mouse away from the initial static corner increases the size of the selection rectangle, while dragging the mouse toward the initial static corner decreases the size of the selection rectangle. A badly placed initial corner can make it difficult and sometimes impossible to select/encapsulate all of the objects. Berkovitz (1994) experimented with a new encirclement technique: the initial area of selection is specified with an encircling gesture and moving the mouse outside of the area enlarges it.

Kids may have trouble double-clicking, and their small hands may have trouble using a three-button mouse (Bederson, Hollan, Druin, Stewart, Rogers, & Proft, 1996). As with adults, children can use point-and-click interfaces more easily than drag-and-drop (Inkpen, 2001; Joiner et al., 1998). Inkpen (2001) noted, "Despite this knowledge, children's software is often implemented to utilize a drag-and-drop interaction style. Bringing solid research and strong results . . . to the forefront may help make designers of children's software think more about the implications of their design choices" (p. 30).

Strommen (1998) noted that since young children cannot reliably tell their left from their right, interfaces for kids should not rely on that distinction. In his Actimates interactive plush toy designs, the toys' left and right legs, hands, and eyes always perform identical functions. More recent interactive soft toys like "Hug & Learn Baby Tad" by Leapfrog (http://www.leapfrog.com) rely in clear visual markings on left and right paws to indicate their distinct functions.

Speech

Speech recognition has intriguing potential for a wide variety of applications for children. O'Hare and McTear (1999) studied use of a dictation program by 12-year-olds and found that they could generate text more quickly and accurately than by typing. They note that dictation automatically avoids some of the errors children would otherwise make, because the recognizer generates correct spelling and capitalization. This is desirable in applications where generating correct text is the goal. If instead, the goal is to teach children to write correctly (and, for example, to capitalize their sentences), then dictation software may be counterproductive.

While O'Hare and McTear (1999) were able to use a standard dictation program with 12-year-olds, Nix, Fairweather, and Adams (1998) noted that speech recognition developed for adults will not work with very young children. In their research on reading tutors for children 5 to 7 years old, they first tried a speech recognizer designed for adults. The recognition rate was only 75%, resulting in a frustrating experience for their subjects. Creating a new acoustic model from the speech of children in the target age range, they were able to achieve an error rate of less than 5%. Further gains were possible by explicitly accounting for common mispronunciations and children's tendency to respond to questions with multiple words where adults would typically provide a one-word answer. Even with the improved acoustic model, the recognizer still made mistakes. To avoid frustrating the children with incorrect feedback, they chose to have the system never tell the child they were wrong. When the system detects what it believes to be a wrong answer, it simply gives the child an easier problem to attempt.

Reading

The written word is a central vehicle for communicating information to humans in human-computer interfaces. Consequently, designing computer technology for children with developing reading skills presents a challenge. Words that are at an appropriate reading level for the target population must be chosen. Larger font sizes are generally preferred. Bernard, Mills, Frank, and McKnown (2001) found that kids 9 to 11 years old prefer 14-point fonts over 12-point. Surprisingly, very little empirical work has been done in this area. Most designers follow the rule of thumb that the younger the child, the larger the font should be.

Designing for preliterate children presents a special challenge. Audio, graphics, and animation must substitute for all functions that would otherwise be communicated in writing. The higher production values required can add significantly to development time and cost. Likewise, visually impaired children pose unique challenges for designers. Audio, tactile, and other sensory interfaces can provide opportunities for visually and otherwise impaired children who cannot use traditional interfaces to interact with computers autonomously (McElligott & van Leeuwen, 2004).

Background Knowledge

Many user interfaces are based on metaphors (Erickson, 1990) from the adult world. Jones (1992) noted that children are less likely to be familiar with office concepts like file folders and in-out boxes. In designing an animation system for kids, Halgren et al. (1995) found many kids to be unfamiliar with the metaphor of a frame-based filmstrip and that of a VCR. It is helpful to choose metaphors that are familiar to kids, though kids often have success in learning interfaces based on unfamiliar metaphors if they are clear and consistent (Schneider, 1996).

Interaction Style

Children's patterns of attention and interaction are quite different from those of adults. Traditional task-oriented analyses of activity may fail to capture the playful, spontaneous nature of children's interactions with technology. For example, when adults are the intended users, designers take great pains to create error messages that are informative and understandable based on the assumption that users want to avoid generating the message again. Hanna, Risden, and Alexander (1997) used a funny noise as an error message and found that the children repeatedly generated the error to hear the noise. Similarly, Halgren et al. (1995) found that children would click on any readily visible feature just to see what would happen, and they might click on it repeatedly if it generated sound or motion in feedback. This behavior was causing young users to be trapped in advanced modes they did not understand. The designers chose to hide advanced functionality in drawers—a metaphor that is familiar to children.

> By hiding the advanced tools, the novice users would not stumble onto them and get lost in their functionality. Rather, only the advanced users who might want the advanced tools would go looking for more options. This redesign allows the product to be engaging and usable by a wider range of ages and abilities. (Halgren et al., 1995)

Resnick and Silverman (2005) went even further in their design principle: "Make it as simple as possible—and maybe even simpler." They warned against "functionality creep" and suggested removing advanced functionality altogether if it is not clearly required to support children's creative efforts.

Children also bring unique interaction styles to online environments; they respond to information they encounter while browsing the web in markedly different ways than adults. In a study of 55 first through fifth graders, the Nielsen Norman

Group (Gilutz & Nielsen, 2002) found that kids were often unable to distinguish between website content and advertisements. Moreover, they rarely scrolled down to find content; instead, they chose to interact with website elements that were immediately visible. When examining a new website, children were willing to hunt for links in the content by "scrubbing the screen" with the mouse instead of relying solely on visual cues (Gilutz & Nielsen, 2002).

Children are more likely than adults to work with more than one person at a single computer. They enjoy doing so to play games (Inkpen, 1997), and they may be forced to do so because of limited resources in school (Stewart, Raybourn, Bederson, & Druin, 1998). Teachers may also create a shared-computer setup to promote collaborative learning. When multiple children work at one machine simultaneously, they need to negotiate sharing control of input devices. Giving students multiple input devices increases their productivity and their satisfaction (Inkpen, 1997; Inkpen, Gribble, Booth, & Klawe, 1995; Stewart et al., 1998). Inkpen et al. (1995) compared two different protocols for transferring control between multiple input devices: give and take. In a give protocol, the user with control clicks the right mouse button to cede it to the other user; in a take protocol, the idle user clicks to take control. In one study with 12-year-olds and another with 9- to 13-year-olds, they found that girls solve more puzzles with a give protocol, but boys are more productive with a take protocol (Inkpen, 1997; Inkpen et al., 1995).

CHILDREN AND THE DESIGN PROCESS

Users play a variety of roles in HCI design processes. Visionary designers, such as Kay (1972) and Papert (1972), began considering the abilities and sensibilities of children in the design of new technologies as early as the 1970s. Today, ethnographic and participatory (Schuler & Namioka, 1993) methods are becoming increasingly common features of the human-centered design toolkit, as HCI designers attempt to deeply understand the practices and preferences of people who will be using new technologies. When designers enter the world of children, and, conversely, when children enter the laboratory, many of the traditional rules change. As we have seen, children are not just little adults; they engage with the world in fundamentally different ways. Naturally, they bring a host of social, emotional, and cognitive elements to the design process that are unfamiliar to designers who are accustomed to working with adults. In this section, we examine new and traditional methods for working with children in a human-centered design process.

Use of Video With Children

Like adults, children may change their behaviors when a video camera is present. Druin (1999) and a design team found in their early work that children tended to "freeze" or "perform" when they saw a video camera in the room. In subsequent work, Druin's team observed that the problems associated with videotaping had more to do with power relationships than with the video cameras themselves. When the children are in control of the cameras, their discomfort decreases (Alborzi et al., 2000). In addition to considering the social impact of using a camera with children, there are also technical difficulties to deal with. Druin's research team found that, even with smaller cameras, it was difficult to capture data in small bedrooms and large public spaces. The sound and speech captured in public spaces was difficult to understand or even inaudible. Finally, it was difficult to know where to place cameras because they did not know where children would sit, stand, or move in the environment. Druin recommended using multiple data sources to capture "messy" design environments with children, including note takers and participant observers in addition to videotaping. Druin also encouraged the design team to use video cameras (along with journal writing, team discussion, and adult debriefing) as a way to record their brainstorming sessions and other design activities.

Goldman-Segall (1996) explained why video data are an important part of ethnographic interviews and observations. When using video, the researcher does not have to worry about remembering or writing down every detail: "She can concentrate fully on the person and on the subtleties of the conversation." The researcher also has access to "a plethora of visual stimuli which can never be 'translated' into words in text," such as body language, gestures, and facial expressions. It is especially important to be able to review the body language of children as they interact with software. Hanna et al. (1997) stated that children's "behavioral signs are much more reliable than children's responses to questions about whether or not they like something, particularly for younger children. Children are eager to please adults, and may tell you they like your program just to make you happy." MacFarlane, Sim, and Horton (2005) suggested that both signs (behaviors) and symptoms (children's direct responses) should be used together to understand children's enjoyments of and abilities to use new technologies. Video is extremely useful in being able to study behavioral signs as the researchers may miss some important signs and gestures during the actual observation or interview.

Instead of using video in its traditional capacity for ethnographic-style observation, some researchers have attempted to capitalize on children's playful treatment of video cameras to elicit articulation about new technologies. In studies using video probes to capture domestic communication patterns, Hutchinson et al. (2003) observed that images are particularly attractive to young people as an entertaining medium for interacting and communicating. During classroom observations of children using math-learning software, Lamberty and Kolodner (2005) encouraged children to engage in "camera talk" with stationary cameras if they wished. Many of the children regularly talked to the camera. This spontaneous behavior revealed both their preferences for using the software and their developing understanding of fractions. Likewise, Iversen (2002) suggested that, by provoking children to verbalize, video cameras provide a communication link between designers and young informants, thereby enriching both the data collected and the design experience.

Methods for Designing and Testing With Kids

In this section, we review a variety of methods for designing with and for children. These methods differ dramatically in the amount of power they grant to children. Some methods encourage us to view children as codesigners with an equal voice in determining design direction, whereas others place children in a more reactive role as evaluators or subjects in laboratory-based usability tests. In practice, designers use methods from different points on this power spectrum depending on the maturity of the project, and often move back and forth between testing with kids and open-ended exploration (Scaife, Rogers, Aldrich, & Davies, 1997).

Druin (2002) unpacked this spectrum of control by describing four different roles that children can play in the design of new technologies: user, tester, informant, and design partner. The most reactive role she described for children in design is user. As users, children interact with existing technologies and have no direct impact on the design of the technology, except in the form of recommendations for future designs. As testers, children are asked to provide feedback about technology in development so that it can be refined before it is released; however, adult designers determine the goals of the technology much earlier. As informants, children play an earlier, more active role in determining the goals and features of new technologies. When children play the role of informant, they interact directly with designers, but ultimately, the designers decide what the children need or want based on observations, interviews, or other data collection methods. Finally, Druin explains that as design partners, children are equal stakeholders in the design process. Although they may not be able to contribute to the development of the technology in equivalent ways, their expertise is equal in importance to that of other contributors to the design process.

The notion of children as design partners will be explored more fully in the section on co-operative inquiry. Methods for including children as informants and design partners borrow from the tradition of participatory design that emerged in the Scandinavian workplace. Participatory design is an "approach towards computer systems design in which the people destined to *use* the system play a critical role in *designing* it" (Schuler & Namioka, 1993, p. xi). With children, this idea is even more important: Since they are physically and cognitively different from adults, their participations in the design process may offer significant insights. Schuler wrote:

> [Participatory Design] assumes that the workers themselves are in the best position to determine how to improve their work and their work life . . . It views the users' perceptions of technology as being at least as important to success as fact, and their feelings about technology as at least as important as what they can do with it. (Schuler & Namioka, 1993, p. xi)

Empowering children in this way and including them in the design process can be difficult due to the traditionally unequal power relationships between kids and adults.

On the other end of the spectrum, methods for including children as users and testers often borrow from the traditional practices of experimental psychology. Usability testing generally takes place in a controlled setting. Sometimes a single design is tested with the goal of improving it; at other times, different design ideas might be compared to establish which ones generate more positive feedback or enable better task completion. Data collection methods like verbal protocol analysis (Ericsson & Simon, 1993) are commonly used and will be further discussed in the section on adapting traditional usability methods for kids.

Cooperative Inquiry

Druin (1999) developed a systematic approach to developing new technologies for children with children; she created new research methods that included children in various stages of the design process. This approach, called "cooperative inquiry," is a combination of participatory design, contextual inquiry, and technology immersion. Children and adults work together on a team as research and design partners. She reiterated the idea that "each team member has experiences and skills that are unique and important, no matter what the age or discipline" (Alborzi et al., 2000, p. 97).

In this model, the research team frequently observes children interacting with software, prototypes, or other devices to gain insight into how child users will interact with and use these tools. When doing these observations, both adult and child researchers observe, take notes, and interact with the child users. During these observations, there are always at least two note-takers and one interactor, and these roles can be filled by either an adult or a child team member. The interactor is the researcher who initiates discussion with the child user and asks questions concerning the activity. If there is no interactor or if the interactor takes notes, the child being observed may feel uncomfortable, like being "on stage" (Druin, 1999). Other researchers have also found that the role of interactor can be useful for members of the design team. Scaife and Rogers (1999) successfully involved children as informants in the development of ECOi, a program that teaches children about ecology. They wanted the kids to help them codesign some animations in ECOi. Rather than just having the software designer observe the children as they played with and made comments about the ECOi prototypes, the software designer took on the role of interactor to elicit suggestions directly. Through these on the fly, high-tech prototyping sessions, they learned that "it was possible to get the software designer to work more closely with the kids and to take on board some of their more imaginative and kid-appealing ideas" (Scaife & Rogers, 1999).

When working as design partners, children are included from the beginning. The adults do not develop all the initial ideas and then later see how the children react to them. The children participate from the start in brainstorming and developing the initial ideas. The adult team members need to learn to be flexible and learn to break away from carefully following their session plans, which is too much like school. Children can perform well in this more improvisational design setting, but the extent to which the child can participate as a design partner depends on his or her age. Children younger than 7 years may have difficulty in expressing themselves verbally and in being self-reflective. These younger children also have difficulty in working with adults to develop new design ideas. Children older than 10 are typically beginning to become preoccupied with preconceived ideas of the way things are supposed to be. In general, it has been found that children ages 7 to 10 years old are the most effective prototyping partners. They are "verbal and self-reflective enough to discuss what they are thinking" (Druin et al., 1999, p. 61), and understand the abstract idea that their low-tech prototypes and designs are going to be turned into technology in the future. They also do not get bogged down with the notion that their designs must be similar to preexisting designs and products.

Through her work with children as design partners, Druin (1999, 2002) has discovered that there are stumbling blocks on the way to integrating children into the design process and to helping adults and children work together as equals. One set of problems deals with the ability of children to express their ideas and thoughts. When the adult and children researchers are doing observations, it is best to allow each group to develop its own style of note taking. Adults tend to take detailed notes, and children tend to prefer to draw cartoons with short, explanatory notes. It is often difficult to create one style of note taking that will suit both groups. Since children may have a difficult time communicating their thoughts to adults, low-tech prototyping is an easy and concrete way for them to create and discuss their ideas. Art supplies such as paper, crayons, clay, and string allow adults and children to work on an equal footing. A problem that arises in practice is that since these tools are childlike, adults may believe that only the child needs to do such prototyping. It is important to encourage adults to participate in these low-tech prototyping sessions.

The second set of problems emerges from the traditionally unequal power relationships between adults and children. In what sense can children be treated as peers? When adults and children are discussing ideas, making decisions, or conducting research, traditional power structures may emerge. In conducting a usability study, the adult researcher might lead the child user through the experiment rather than allowing the child to explore freely on his or her own. In a team discussion, the children may act as if they are in a school setting by raising their hands to speak. Adults may even inadvertently take control of discussions. Is it sensible to set up design teams where children are given equal responsibilities to those of adult designers? Getting adults and children to work together as a team of equals is often the most difficult part of the design process. It is to be expected that it may take a while for a group to become comfortable and efficient when working together. It can take up to six months for an "intergenerational design team to truly develop the ability to build upon each other's ideas" (Druin et al., 2001). To help diffuse such traditional adult-child relationships, adults are encouraged to dress casually, and there always should be more than one adult and more than one child on a team. A single child may feel outnumbered by the adults, and a single adult might create the feeling of a school environment where the adult takes on the role of teacher. Alborzi et al. (2000) started each design session with 15 minutes of snack time, where adults and children can informally discuss anything. This helps both adults and children get to know each other better as "people with lives outside of the lab" (Alborzi et al., 2000) and to improve communication within the group.

Scaife et al. (1997) identified aspects of working with children in the role of informant that require special attention. They found that, when working in pairs, children feel less inhibited about telling strange adults what they were thinking. Other researchers have also found that pairing children, especially with friends, can help ease discomfort (Dindler, Eriksson, Iversen, Ludvigsel, & Lykke-Olesen, 2005; Als, Jensen, & Skov, 2005). Scaife et al. (1997) cautioned that adults also need to become comfortable in the role of facilitator and should take care not to intervene too quickly if children's discussions wander.

In addition to the social challenges associated with mixing adults and children as equal design partners, kids do not always know how to collaborate well with one another in the first place. Because collaborating on a design project is often a novel experience for children, organizing the activities (without imposing too rigid a structure) can help create productive sessions. For example, Guha, Druin, Chipman, Fails, Simms, and Farber (2005) described a technique to support collaboration among kids and adults during cooperative inquiry sessions called "mixing ideas." First, kids generate ideas in a one-on-one session with an adult facilitator, then they work in small groups to integrate these ideas, and finally they work in larger groups until the whole group is finally working together.

Although there have been many successes in having children participate as design and research partners in the development of software, there are still many questions to be answered about the effectiveness of this approach. Scaife and Rogers (1999) attempted to address many of the questions and problems faced when working with children in their work on informant design. The first question deals with the multitude of ideas and suggestions produced by children. Children say outrageous things. How do you decide which ideas are worthwhile? When do you stop listening? The problem of selection is difficult since, in the end, the adult decides which ideas to use and which ideas to ignore. Scaife and Rogers suggested creating a set of criteria to:

> Determine what to accept and what not to accept with respect to the goals of the system . . . You need to ask what the trade-offs will be if an idea or set of ideas are implemented in terms of critical 'kid' learning factors: that is, how do fun and motivation interact with better understanding? (Scaife & Rogers, 1999, pp. 46–47)

In addition to deciding which of the children's ideas to use, there is also the problem of understanding the meaning behind what the child is trying to say. Adults tend to assume that they can understand what kids are getting at, but kid talk is not adult talk. It is important to remember that children have "a different conceptual framework and terminology than adults" (Scaife & Rogers, 1999, p. 47).

Another problem with involving children, particularly with the design of educational software, is that "children can't discuss learning goals that they have not yet reached themselves" (Scaife & Rogers, 1999, p. 30). Can children make effective contributions about the content and the way they should be taught, something which adults have always been responsible for? Adults have assumptions about what is an effective way to teach children. Kids tend to focus on the fun aspects of the software rather than the educational agenda. A mismatch of expectations may exist if kids are using components of the software in unanticipated ways. Involving children in the design and evaluation process may help detect where these mismatches occur in the software.

Adapting Usability Evaluation Methods

HCI practices have evolved to address usefulness, enjoyability, and other measures of design success; however, usability remains a fundamental concern for HCI designers. Although efficiency and task completion are often not central to kids' goals in using technology, usability problems can create barriers to achieving other goals. For example, much research done to date has focused on designing educational software, and evaluation is primarily of learning outcomes, not usability. However, usability is a prerequisite for learning. In student projects in Georgia Tech's graduate class "Educational Technology: Design and Evaluation," many student designers are never able to show whether the educational design of their software is successful. What they find instead is that usability problems intervene, and they are unable even to begin to explore pedagogical efficacy. If children cannot use educational technology effectively, they certainly will not learn through the process of using it. MacFarlane et al. (2005) found that measurements of usability and "fun" were significantly correlated in studies of educational software for science. Usability is similarly important for entertainment, communications, and other applications. Many researchers have explored the effectiveness of traditional usability methods with children. In this section, we examine comparative assessments of usability methods and review findings and recommendations.

Traditional usability testing. Several guidelines developed for work with adults become more important when applied to children. For example, when children are asked to work as testers, it is important to emphasize that it is the software that is being tested, not the participant (Rubin, 1994). Children might become anxious at the thought of taking a test, and test taking may conjure up thoughts of school. The researcher can emphasize that even though the child is participating in a test, the child is the tester and not the one being tested (Hanna et al., 1997). Rubin recommends that you show the participant where video cameras are located; let them know what is behind the one-way mirror and whether people will be watching. With children, showing them behind the one-way mirrors and around the lab gives them "a better sense of control and trust in you" (Hanna et al., 1997, p. 12).

Markopoulos and Bekker (2002) described characteristics of kids that can impact the process and outcome of usability testing:

- Children's capacity to verbalize thoughts is still developing.
- Personality may impact both kid's willingness to speak up to adults and their motivation to please authority figures.
- The capacity to concentrate is variable among kids.
- Young children are still developing the capacity for abstract and logical thinking; they may differ in cognitive ability such as remembering several items at once.
- The ability to monitor goal-directed performance develops throughout childhood and adolescence.
- Some ages may have more pronounced gender differences than others may.
- With small children, basic motor skills ability may be a barrier to effective evaluation if kids cannot use prototypes with standard input devices.

Hanna et al. (1997) developed a set of guidelines for laboratory-based usability testing with children:

- The lab should be made a little more child friendly by adding some colorful posters but avoid going overboard as too many extra decorations may become distracting to the child.
- Try to arrange furniture so that children are not directly facing the video camera and one-way mirror, as the children may choose to interact with the camera and mirror rather than the doing the task.
- Children should be scheduled for an hour of lab time. Preschoolers will generally only be able to work for 30 minutes but will need extra time to play and explore. Older children will become tired after an hour of concentrated computer use, so if the test will last longer than 45 minutes, children should be asked if they would like to take a short break at some point during the session.
- Hanna et al. (1997) suggested that you "explain confidentiality agreements by telling children that designs are 'top-secret'" (p. 12). Parents should also sign the agreements, since they will inevitably also see and hear about the designs.
- Children up to seven or eight years old will need a tester in the room with them for reassurance and encouragement. They may become agitated from being alone or following directions from a loudspeaker. If a parent will be present in the room with the child, it is important to explain to the parent that he or she should interact with the child as little as possible during the test. Older siblings should stay in the observation area or a separate room during the test, as they may eventually be unable to contain themselves and start to shout out directions.
- Hanna et al. (1997) suggested that you should "not ask children if they want to play the game or do a task—that gives them the option to say no. Instead use phrases such as 'Now I need you to,' 'Let's do this,' or 'It's time to.'"

Think/talk aloud. An important method for collecting usability data with adults is think-aloud protocols. Think-aloud protocols in HCI research are related to verbal protocol analysis methods in psychology, in which subjects are asked to describe what they are thinking about and paying attention to while they complete some set of tasks (Ericsson & Simon, 1993). In usability tests, think-aloud methods are generally used in concert with direct observation (Nielsen, 1993). Researchers who have used this method with children have observed that children may make very few comments during testing (Donker & Reitsma, 2004). In some cases, they seem to have difficulty with concurrent verbalization—verbalizing thoughts while they complete tasks. The cognitive load associated with learning and executing task itself might interfere with kids' abilities to talk about it (Hoysniemi, Hamalainen, & Turkki, 2003).

Despite potential obstacles, it has been demonstrated that verbal comments from children can play an important role in identifying usability problems. Donker and Reitsma (2004) reported that, although children produced fewer comments, those few comments provided important information about the severity of usability problems that were identified by direct observation. Likewise, other work suggested that, although using think alouds with kids may result in fewer utterances than other approaches, it can be used to generate useful usability data with both older kids aged 8 to 14 (Donker & Markopoulos, 2002; Baauw & Markopoulos, 2004) and younger children aged 6 to 7 (van Kesteren, Bekker, Vermeeren, & Lloyd, 2003).

Active intervention is closely related to think-aloud protocols but involves investigators asking planned questions to encourage testers to reflect aloud on actions at specific points while completing a task. In a small comparative study with kids ages six and seven, van Kesteren et al. (2003) found that active intervention elicited the most comments when compared to think alouds, posttask/retrospection, codiscovery, peer tutoring, and traditional usability testing.

Post-task interviews. In posttask interviews, testers are asked to describe their experiences after they have finished using a new technology to complete a set of tasks. In some cases, video data are reviewed with the participant to evoke comments. This kind of retrospective verbal protocol emerged from the same tradition as think alouds (Ericsson & Simon, 1993). Van Kesteren et al. (2003) raised the question of whether young kids' limited capacities to hold in memory several concepts at once and still-developing abilities to engage in abstract thought limit their abilities to accurately recall and recount past actions. They found that some kids age six and seven were able to recall past actions and describe the ways in which their understanding changed. They note that keeping things interesting is important; children become bored with reviewing videos unless the tasks themselves are engaging to watch. In studies with kids ages 9 to 11, Baauw and Markopoulos (2004) determined that posttask interviews alone revealed fewer usability problems than think alouds; however, when combined with data from observations, which is standard practice, there was no significant difference between the problems that the two methods revealed.

Codiscovery. Codiscovery exploration is a usability method that is used to understand users' experiences and perceptions of new product designs, especially those that may be unfamiliar. In codiscovery sessions, two users who know one another work together to perform a set of tasks using the product. The goal of using two acquainted users is to encourage them to talk about the problems they encounter and their perceptions of the product in the natural course of collaborating on a task instead of relying on a single user's verbal performance for an experimenter (Kemp & van Gelderen, 1996). First, the two users are asked to figure out what a product does and compare it other products they know about. Next, they are asked to collaborate on a set of specific tasks using the product. Finally, a discussion period allows designers to ask about observed problems and behaviors; in addition, participants can ask questions about the design and the intended purpose of the product. Van Kesteren et al. (2003) found that using co-discovery with kids ages six and seven can be difficult because they often attempt to complete tasks individually. Even when seated next to one another, two children may not interact at all, resulting in very few comments about the product being tested. When compared with traditional usability tests, think alouds, posttask interviews, peer tutoring and active intervention, codiscovery was found to elicit the fewest comments from kids (van Kesteren et al., 2003).

Peer tutoring. Peer tutoring is a method for usability testing that was developed to capitalize on the ways that children interact with one another in natural, playful settings. When children play together, they regularly teach one another games and invent rules of play. Hoysniemi et al. (2003) explained, "One definition of the usability of a children's software application is that a child is able and willing to teach other children how to use it" (p. 209). Instead of relying on task completion in a lab, peer tutoring is an approach to usability testing that allows kids to engage in exploratory and playful interactions in a naturalistic setting. Peer tutoring involves first helping one or more kids to develop expertise using a piece of software and then asking them to teach other kids how to use it. By observing, recording, and analyzing interactions between tutors and tutees, it is possible to identify usability problems in software as the kids attempt to teach it to one another. Hoysniemi et al. (2003) pointed out that, although it can be useful, the peer-tutoring approach requires time, training, and careful implementation to be effective.

GENRES OF TECHNOLOGY FOR KIDS

Technology for kids falls into two broad categories: education and entertainment. When game companies try to mix these genres, they may use the term *edutainment.* New products for kids increasingly include specialized hardware as well as software.

Entertainment

Designers of games and other entertainment software rarely write about how they accomplish their jobs. Talks are presented each year at the Game Developer's Conference (http://www.gdconf.com), and some informal reflections are gathered as conference proceedings. Attending the conference is recommended for people who wish to learn more about current issues in game design. *Game Developer* magazine is the leading publication with reflective articles on the game design process.

Game designers are usually gamers themselves and often end up simply designing games that they themselves would like to play. This simple design technique is easy and requires little if any background research with users. Because most game designers have traditionally been male, this approach allowed them to appeal quite effectively to a core gaming audience: young men and teenage boys. However, female designers are becoming more common on design teams and gaming companies are increasingly recognizing that people outside the typical gamer stereotype represent a large potential market for their products. Designing for teenagers is relatively easy. As we have seen, designing for very young children presents substantial challenges. The younger your target audience, the more they should be tightly connected to every stage of the design process.

Brenda Laurel pioneered the use of careful design methods for nontraditional game audiences in her work with the company Purple Moon in the mid-1990s. Laurel aimed to develop games that appeal to preteen girls both to tap this market segment and to give girls an opportunity to become fluent with technology. Many people believe that use of computer game leads to skills that later give kids advantages at school and work. Through extensive interviews with girls in their target age range, Purple Moon was able to create successful characters and game designs. However, the process was so time consuming and expensive that the company failed to achieve profitability fast enough to please its investors. The company closed in 1999, and its characters and games were sold to Mattel. Purple Moon perhaps did more research than was strictly necessary, particularly because their area was so new. The broader lesson is that the game industry typically does not budget for needs analysis and iterative design early in the design process. Playtesting and quality assurance typically take place relatively late in the design cycle. Designers contemplating incorporating research early in their design process must consider the financial costs.

Oosterholt et al. (1996) described several design constraints that are specific to the design of products for children. First, they suggested that trying to be fashionable might result in products that kids quickly perceived as outdated. They also pointed out that fun is just as important to measure as usability, that measurements of fun should be shared with development teams, and moreover, that the product should grow with users over time and continue to be fun long after kids have learned to use it.

Game designer Carolyn Miller (1998) highlighted seven mistakes (or "kisses of death") commonly made by people trying to design games for kids:

"Death kiss #1: Kids love anything sweet"
Miller (1998) wrote, "Sweetness is an adult concept of what kids should enjoy." Only very young children will tolerate it. Humor and good character development are important ingredients. Do not be afraid to use off-color humor or to make something scary.

"Death kiss #2: Give 'em what's good for 'em"
Miller advised, "Don't preach, don't lecture, and don't talk down—nothing turns kids off faster."

"Death kiss #3: You just gotta amuse 'em"
Miller wrote, "Don't assume that just because they are little, they aren't able to consume serious themes."

"Death kiss #4: Always play it safe!"
Adult games often rely on violence to maintain dramatic tension. Since you probably will not want to include this in your game for kids, you will need to find other ways to maintain dramatic tension. Do not let your game become bland.

"Death kiss #5: All kids are created equal"
Target a specific age group, and take into consideration humor, vocabulary, skill level, and interests. If you try to design for everyone, your game may appeal to no one.

"Death kiss #6: Explain everything"
In an eagerness to be clear, some people overexplain things to kids. Kids are good at figuring things out. Use as few words as possible, and make sure to use spoken and visual communication as much as possible.

"Death kiss #7: Be sure your characters are wholesome!"
Miller warned that if every character is wholesome, the results are predictable and boring. Characters need flaws to have depth. Miller identified a number of common pitfalls in assembling groups of characters. It is not a good idea to take a "white bread" approach, in which everyone is White and middle class. On the other end of the spectrum, it is also undesirable to take a "lifesaver approach" with one character for each ethnicity. Finally, you also need to avoid an "off-the-shelf" approach, in which each character represents a stereotype: "You've got your beefy kid with bad teeth; he's the bully. You've got the little kids with glasses; he's the smart one." Create original characters that have depth and have flaws that they can struggle to overcome (Miller, 1998).

Education

To design educational software, we must expand the concept of user-centered design (UCD) to one of learner-centered design (LCD; Soloway, Guzdial, & Hay, 1994). There are several added steps in the process:

- Needs analysis
  - For learners
  - For teachers
- Select pedagogy
- Select media/technology
- Prototype
  - Core application
  - Supporting curricula
  - Assessment strategies
- Formative evaluation
  - Usability
  - Learning outcomes
- Iterative design
- Summative evaluation
  - Usability
  - Learning outcomes

In our initial needs analysis, for software to be used in a school setting, we need to understand not just learners but also teachers. Teachers have heavy demands on their time and are held accountable for their performances in ways that vary between districts and between election years.

Once we understand our learners and teachers needs, we need to select an appropriate pedagogy—an approach to teaching and learning. For example, behaviorism views learning as a process of stimulus and reinforcement (Skinner, 1968). Constructivism sees learning as a process of active construction of knowledge through experience. A social-constructivist perspective emphasizes learning as a social process (Newman, Griffin, & Cole, 1989). (A full review of approaches to pedagogy is beyond the scope of this chapter.)

Next, we are ready to select the media we will be working with, matching their affordances to our learning objectives and pedagogical approach. Once the prototyping process has begun, we need to develop not just software or hardware, but (for applications to be used in schools) also supporting curricular materials and assessment strategies.

Assessment should not be confused with evaluation. The goal of assessment is to judge an individual student's performance. The goal of evaluation is to understand to what extent our learning technology design is successful. An approach to assessing student achievement is an essential component of any school-based learning technology. For both school and free time use, we need to design feedback mechanisms so that learners can be aware of their progress. It is also important to note whether learners find the environment motivating. Does it appeal to all learners or to specific gender, learning style, or interest groups?

As in any HCI research, educational technology designers use formative evaluation to understand informally what needs improvement in their learning environments, and guide the process of iterative design. Formative evaluation must pay attention first to usability, and second to learning outcomes. If students cannot use the learning hardware or software, they certainly will not learn through its use. Once it is clear that usability has met a minimum threshold, designers then need to evaluate whether learning outcomes are being met. After formative evaluation and iterative design are complete, a final summative evaluation serves to document the effectiveness of the design and justify its use by learners and teachers. Summative evaluation must similarly pay attention to both usability and learning outcomes.

A variety of quantitative and qualitative techniques are commonly used for evaluation of learning outcomes (Gay & Airasian, 2000). Most researchers use a complementary set of both quantitative and qualitative approaches. Demonstrating educational value is challenging, and research methods are an ongoing subject of research.

This represents an idealized learner-centered design process. Just as many software design projects do not in reality follow a comprehensive UCD process, many educational technology projects do not follow a full LCD process. LCD is generally substantially more time consuming than UCD. While in some cases it may be possible to collect valid usability data in a single session, learning typically takes place over longer periods. To get meaningful data, most classroom trials take place over weeks or months. Furthermore, classroom research needs to fit into the school year at the proper time. If you are using Biologica (Hickey, Kindfield, Horwitz, & Christie, 2000) to teach about genetics, you need to wait until it is time to cover genetics that school year. You may have only one or two chances per year to test your educational technology. It frequently takes many years to complete the LCD process. In the research community, one team may study and evolve one piece of educational technology over many years. In a commercial setting, educational products need to get to market rapidly, and this formal design process is rarely used.

Genres of Educational Technology

Taylor (1980) divided educational technology into three genres:

- Computer as tutor
- Computer as tool
- Computer as tutee

Suppose that we are learning about acid rain. If the computer is serving as tutor, it might present information about acid rain and ask the child questions to verify the material was understood. If the computer is a tool, the child might collect data about local acid rain and input that data into an ecological model to analyze its significance. If the computer is a tutee, the child might program his or her own ecological model of acid rain.

With the advent of the Internet, we must add a fourth genre:

- Computer-supported collaborative learning (CSCL)

In a CSCL study of acid rain, kids from around the country might collect local acid rain data, enter it into a shared database, analyze the aggregate data, and talk online with adult scientists who study acid rain. This is in fact the case in the NGS-TERC Acid Rain Project (Tinker, 1993). See Table 40.1 for an overview of genres of children's software.

Computer as Tutor

In most off the shelf educational products, the computer acts as tutor. Children are presented with information and then quizzed on their knowledge. This approach to education is grounded in behaviorism (Skinner, 1968). It is often referred to as "drill and practice" or "computer-aided instruction" (CAI).

TABLE 40.1. Genres of Children's Software

Genre	Description
Entertainment	Games created solely for fun and pleasure.
Educational	Software created to help children learn about a topic using some type of pedagogy—an approach to teaching and learning.
Computer as Tutor	Often referred to as "drill and practice" or "computer-aided instruction" (CAI), this approach is grounded in behaviorism. Children are presented with information and then quizzed on their knowledge.
Computer as Tool	The learner directs the learning process, rather than being directed by the computer. This approach is grounded in constructivism, which sees learning as an active process of constructing knowledge through experience.
Computer as Tutee	Typically, the learner uses construction kits to help reflect upon what he or she learned through the process of creation. This approach is grounded in constructivism and constructionism.
Computer-supported Collaborative learning (CSCL)	Children use the Internet to learn from and communicate with knowledgeable members of the adult community. Children can also become involved in educational online communities with children from different geographical regions. This approach is grounded in social constructivism.
Edutainment	A mix of the entertainment and educational genres.

The computer tracks student progress and repeats exercises as necessary.

Researchers with a background in artificial intelligence have extended the drill and practice approach to create intelligent tutoring systems. Such systems try to model what the user knows and tailor the problems presented to an individual's needs. Many systems explicitly look for typical mistakes and provide specially prepared corrective feedback. For example, suppose a child adds 17 and 18 and gets an answer of 25 instead of 35. The system might infer that the child needs help learning to carry from the ones to the tens column and present a lesson on that topic. One challenge in the design of intelligent tutors is in accurately modeling what the student knows and what their errors might mean.

Byrne, Anderson, Douglass, and Matessa (1999) experimented with using eye tracking to improve the performance of intelligent tutors. Using an eye tracker, the system can tell whether the student has paid attention to all elements necessary to solve the problem. In early trials with the eye tracker, they found that some of the helpful hints the system was providing to the user were never actually read by most students. This helped guide their design process. They previously focused on how to improve the quality of hints provided; however, that is irrelevant if the hints are not even being read (Byrne et al., 1999).

An interesting variation on the traditional 'computer as tutor' paradigm for very young children is the Actimates line interactive plush toys. Actimates Barney and other characters lead children in simple games with educational value, like counting exercises. The tutor is animated and anthropomorphized. The embodied form lets young children use the skills they have in interacting with people to learn to interact with the system, enhancing both motivation and ease of use (Strommen, 1998; Strommen & Alexander, 1999).

Computer as Tool

When the computer is used as a tool, agency shifts from the computer to the learner. The learner is directing the process, rather than being directed. This approach is preferred by constructivist pedagogy, which sees learning as an active process of constructing knowledge through experience. The popular drawing program Kid Pix is an excellent example of a tool customized for kids' interests and needs. Winograd (1996) commented that Kid Pix's designer Craig Hickman "made a fundamental shift when he recognized that the essential functionality of the program lay not in the drawings that it produced, but in the experience for the children as they used it" (p. 60). For example, Kid Pix provides several different ways to erase the screen—including having your drawing explode or be sucked down a drain.

Simulation programs let learners try out different possibilities that would be difficult or impossible in real life. For example, Biologica (an early version was called "Genscope") allows students to learn about genetics by experimenting with breeding cartoon dragons with different inherited characteristics like whether they breathe fire or have horns (Hickey et al., 2000). Model-it lets students try out different hypotheses about water pollution and other environmental factors in a simulated ecosystem (Soloway et al., 1996).

The goal of such programs is to engage students in scientific thinking. The challenge in their designs is how to get students to think systematically and not to simply try out options at random. Programs like Model-It provide the student with scaffolding. Initially, students are given lots of support and guidance. As their knowledge evolves, the scaffolding is faded, allowing the learner to work more independently (Guzdial, 1994; Soloway et al., 1994).

Computer as Tutee

Papert (1992) commented that much computer-aided instruction is "using the computer to program the child" (p. 163). Instead, he argued that the child should learn to program the computer and through this process gain access to new ways of thinking and understanding the world. Early research argued that programming would improve children's general cognitive skills, but empirical trials produced mixed results (Clements, 1986; Clements & Gullo, 1984; Pea, 1984). Some researchers argued that the methods of these studies are fundamentally flawed, because the complexity of human experience cannot be reduced to pretests and posttests (Papert, 1987). The counterargument is that researchers arguing that technology has a transformative power need to back up their claims with evidence of some form, whether quantitative or qualitative (Pea, 1987; Walker, 1987). More recently, the debate has shifted to the topic of technological fluency. As technology increasingly surrounds our everyday lives, the ability to use it effectively as a tool becomes important for children's successes in school and later in the workplace (Resnick & Rusk, 1996).

In the late 1960s, Feurzeig (1996) and colleagues at BBN invented Logo, the first programming language for kids. Papert (1980) extended Logo to include turtle graphics, in which kids learn geometric concepts by moving a turtle around the screen. A variety of programming languages for kids have been developed over subsequent years, including Starlogo (Resnick, 1994), Boxer (diSessa & Abelson, 1986), Stagecast (Cypher & Smith, 1995), Agentsheets (Repenning & Fahlen, 1993), MOOSE (Bruckman, 1997), and Squeak (Guzdial & Rose, 2001). Lego Mindstorms (originally "Lego/Logo") is a programmable construction kit with physical as well as software components (Martin & Resnick, 1993). Another programmable tool bridging the gap between physical constructions and representations on the screen is Hypergami, a computer-aided design tool for origami developed at the University of Colorado at Boulder. Students working with Hypergami learn about both geometry and art (Eisenberg, Nishioka, & Schreiner, 1997).

In most design tools, the goal is to facilitate the creation of a product. In educational construction kits, the goal instead is what is learned through the process of creation. So what makes a good construction kit? In an *Interactions* article entitled "Pianos, Not Stereos: Creating Computational Construction Kits," Resnick, Bruckman, and Martin (1996) discussed the art of designing construction kits for learning (constructional design):

> 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

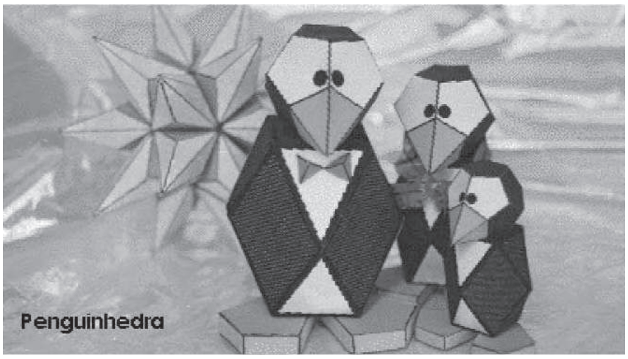

FIGURE 40.2. Penguins created using Hypergami.

"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.

Bruckman (2000) added a third design principle:

- *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 everyday activities.

Resnick and Silverman (2005) suggested several more design principles for creating construction kits for kids. Some, such as "iterate, iterate—then iterate again," are familiar mantras for HCI designers. Others may be less familiar:

- Low Floor and Wide Walls
  If a technology has a low floor, it means that it is easy for novices to begin using it. Wide Walls suggest a wide range of possible areas of design and exploration. Construction kits define "a place to explore, not a collection of specific activities."
- Make Powerful Ideas Salient—Not Forced
  When designing toward specific learning goals, construction kits should make these ideas visible and useful in design activities rather than imposing the ideas on students as a predetermined solution.
- Support Many Paths, Many Styles
  Kids approach problems in different ways, it is important to support a variety of design approaches in a construction kit.
- Make it as Simple as Possible—and Maybe Even Simpler
  Constraints can be the designer's best friend. Limited functionality sometimes wins out over designs that are more sophisticated because simplicity allows kids to find creative new ways to use a product.
- Choose Black Boxes Carefully
  This principle is related to the previous one; deciding when to reveal complexity and when to conceal it is a difficult question. Resnick and Silverman (2005) suggested that the simplest choice is often the best one.
- A Little Bit of Programming Goes a Long Way
  Because programming is the fundamental mode of construction with computers, designers of construction kits for kids often include some programming functionality. Focusing on powerful, simple commands that kids can do well is often the best way to support a diverse range of activities.
- Give People What They Want—Not What They Ask For
  Observations of kids can often tell designers more than their direct answers to questions. Kids may ask for unrealistic features or may not know themselves why they are having difficulty completing a task.
- Invent Things That You Would Want to Use Yourself
  Although they caution against overgeneralizing one's own personal likes and dislikes, Resnick and Silverman (2005) proposed that the most respectful approach to designing for kids is to create something that the designer herself finds enjoyable.

Computer-Supported Collaborative Learning (CSCL)

Most tools for learning have traditionally been designed for one child working at the computer alone. However, learning is generally recognized to be a social process (Newman et al., 1989). With the advent of the Internet came new opportunities for children to learn from one another and from knowledgeable members of the adult community. This field is called "Computer-Supported Collaborative Learning" (CSCL; Koschmann, 1996).

CSCL research can be divided into four categories:

1. Distance education
   Students attempt to use online environments in ways that emulate a traditional classroom.
2. Information retrieval
   Research projects in which students use the Internet to find information.
3. Information sharing

Students debate issues with one another. One of the first such tools was the Computer-Supported Intentional Learning Environment (CSILE), a networked discussion tool designed to help students engage in thoughtful debate as a community of scientists does (Scardamalia & Bereiter, 1994). They may also collect scientific data and share it with others online. In the "One Sky, Many Voices" project, students learn about extreme weather phenomena by sharing meteorological data they collect with other kids from around the world, and also by talking online with adult meteorologists (Songer, 1996). In the Palaver Tree Online project, kids learn about history by talking online with older adults who lived through that period of history (Ellis & Bruckman, 2001). A key challenge in the design of information sharing environments is how to promote serious reflection on the part of students (Guzdial, 1994; Kolodner & Guzdial, 1996).

Technological Samba Schools

In *Mindstorms*, Papert (1980) had a vision of a "technological samba school." At samba schools in Brazil, a community of people of all ages gather together to prepare a presentation for carnival. "Members of the school range in age from children to grandparents and in ability from novice to professional. But they dance together and as they dance everyone is learning and teaching as well as dancing. Even the stars are there to learn their difficult parts" (Papert, 1980). People go to samba schools not just to work on their presentations, but also to socialize and be with one another. Learning is spontaneous, self-motivated, and richly connected to popular culture. Papert imagined a kind of technological samba school where people of all ages gather together to work on creative projects using computers. The Computer Clubhouse is an example of such a school in a face-to-face setting (Resnick & Rusk, 1996). MOOSE Crossing is an Internet-based example (Bruckman, 1998). A key challenge in the design of such environments is how to grapple with the problem of uneven achievement among participants. When kids are allowed to work or not work in a self-motivated fashion, typically some excel while others do little (Elliott, Bruckman, Edwards, & Jensen, 2000).

Child Safety Online

One challenge in the design of Internet-based environments for kids is the question of safety. The Internet does contain information that is sexually explicit, violent, and racist. Typically, such information does not appear unless one is looking for it; however, it is unusual but possible to stumble across it accidentally. Filtering software blocks access to useful information as well as harmful (Schneider, 1997). Furthermore, companies that make filtering software often fail to adequately describe how they determine what to block, and they may have unacknowledged political agendas that not all parents will agree with. Resolving this issue requires a delicate balance of the rights of parents, teachers, school districts, and children (Electronic Privacy Information Center, 2001). Another danger for kids online is the presence of sexual predators and others who wish to harm children. While such incidents are rare, it is important to teach kids not to give out personal information online such as their last names, addresses, or phone numbers. Kids who wish to meet online friends to friend face to face should do so by each bringing a parent and meeting in a well-populated public place like a fast-food restaurant. A useful practical guide "Child Safety on the Information Superhighway" is available from the Center for Missing and Exploited Children (http://www.missingkids.com). Educating kids, parents, and teachers about online safety issues is an important part of the design of any online software for kids.

CONCLUSION

To design for kids, we must have a model of what kids are and what we would like them to become. Adults were once kids. Many are parents. Some are teachers. We tend to think that we know kids—who they are, what they are interested in, and what they like. However, we do not have as much access to our former selves as many would like to believe. Furthermore, it is worth noting that our fundamental notions of childhood are in fact culturally constructed and change over time. Calvert (1992) wrote about the changing notion of childhood in America, and the impact it has had on artifacts designed for children and child rearing:

> In the two centuries following European settlement, the common perception in America of children changed profoundly, having first held to an exaggerated fear of their inborn deficiencies, then expecting considerable self-sufficiency, and then, after 1830, endowing young people with an almost celestial goodness. In each era, children's artifacts mediated between social expectations concerning the nature of childhood and the realities of child-rearing: before 1730, they pushed children rapidly beyond the perceived perils of infancy, and by the nineteenth century they protected and prolonged the perceived joys and innocence of childhood. (p. 8)

While Calvert (1992) reflected on the design of swaddling clothes and walking stools, the same role is played by new technologies for kids like programmable Legos and drill and practice arithmetic programs: These artifacts mediate between our social expectations of children and the reality of their lives. If you believe that children are unruly and benefit from strong discipline, then you are likely to design CAI. If you believe that children are creative and should not be stifled by adult discipline, then you might design an open-ended construction kit like Logo or Squeak. In designing for kids, it is crucial to become aware of one's own assumptions about the nature of childhood. Designers should be able to articulate their assumptions, and be ready to revise them based on empirical evidence.

References

Alborzi, H., Druin, A., Montemayor, J., Platner, M., Porteous, J., Sherman, L., et al. (2000). Designing StoryRooms: Interactive Storytelling Spaces for Children. *Proceedings of the Symposium on Designing interactive systems: Processes, practices, methods, and techniques,* Brooklyn, NY, 95–104.

Als, B. S., Jensen, J. J., & Skov, M. B. (2005). Comparison of think-aloud and constructive interaction in usability testing with children. *Proceedings of the 2005 Conference on Interaction Design and Children*, Boulder, CO, 9–16.

BBC. (1997). *Teletubbies Press Release*.

Baauw, E., & Markopoulos, P. (2004). A Comparison of think-aloud and post-task interview for usability testing with children. *Proceedings of the 2004 Conference on Interaction Design and Children*, College Park, MD, 115–116.

Bederson, B., Hollan, J., Druin, A., Stewart, J., Rogers, D., & Proft, D. (1996). Local Tools: An Alternative to Tool Palettes. *Proceedings of the ACM Symposium on User Interface Software and Technology*, Seattle, WA, 169–170.

Berkovitz, J. (1994). Graphical Interfaces for Young Children in a Software-based Mathematics Curriculum. *Proceedings of the ACM Conference on Human Factors in Computing Systems: Celebrating Interdependence*, Boston, MA, 247–248.

Bernard, M.., Mills, M., Frank, T., & McKnown, J. (2001, Winter). Which fonts do children prefer to read online? *Software Usability Research Laboratory* (*SURL*). Retrieved March 16, 2007, from psychology .wichita.edu/surl/usabilitynews/w3/fontJR.htm

Bruckman, A. (1997). *MOOSE Crossing: Construction, community, and learning in a networked virtual world for kids.* Unpublished doctoral dissertation, MIT.

Bruckman, A. (1998). Community support for constructionist learning. *Computer Supported Cooperative Work, 7*, 47–86.

Bruckman, A. (2000). Situated support for learning: Storm's weekend with Rachael. *Journal of the Learning Sciences, 9*(3), 329–372.

Byrne, M. D., Anderson, J. R., Douglass, S., & Matessa, M. (1999). Eye tracking the visual search of click-down menus*. Proceedings of the ACM Conference on Human Factors in Computing Systems: the CHI is the limit*, Pittsburgh, PA, 402–409.

Calvert, K. (1992). *Children in the House: The Material Culture of Early Childhood, 1600–1900*. Boston: Northeastern University Press.

Clements, D. H. (1986). Effects of Logo and CAI Environments on Cognition and Creativity. *Journal of Educational Psychology, 78*(4), 309–318.

Clements, D. H., & Gullo, D. F. (1984). Effects of Computer Programming on Young Children's Cognition. *Journal of Educational Psychology, 76*(6), 1051–1058.

Cypher, A., & Smith, D. C. (1995). End user programming of simulations. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, Denver, CO, 27–34.

Davenport, A., & Wood, A. (1997). *TeleTubbies FAQ*. BBC Education. Retrieved date, from http://www.bbc.co.uk/cbeebies/teletubbies/grownups/faq.shtml

Dindler, C., Eriksson, E., Iversen, O. S., Ludvigsel, M., & Lykke-Olesen, A. (2005). Mission from Mars: A method for exploring user requirements for children in a narrative space. *Proceedings of the 2005 Conference on Interaction Design and Children*, Boulder, CO, 40–47.

diSessa, A. A., & Abelson, H. (1986). Boxer: A reconstructible computational medium. *Communications of the ACM, 29*(9), 859–868.

Donker, A., & Markopoulos, P. (2002). A comparison of think-aloud, questionnaires and interviews for testing usability with children. *Proceedings of HCI 2002*, location, 305–316.

Donker, A., & Reitsma, P. (2004). Usability testing with young children. *Proceedings of the Conference on Interaction Design and Children*, College Park, MD, 43–48.

Dourish, P (2001). *Where the action is: The foundations of embodied interaction*. Cambridge, MA: MIT Press.

Druin, A. (1999). Cooperative inquiry: developing new technologies for children with children. *Proceedings of the ACM Conference on Human Factors in Computing Systems: the CHI is the limit*, Pittsburgh, PA, 592–599.

Druin, A. (2002). The role of children in the design of new technology. *Behaviour & Information Technology, 21*(1), 1–25.

Druin, A., Bederson, B., Hourcade, J. P., Sherman, L., Revelle, G., Platner, M., et al. (2001). Designing a digital library for young children: An intergenerational partnership*. Proceedings of the Joint Conference on Digital Libraries*, Roanoke, VA, 398–405.

Eisenberg, M., Nishioka, A., & Schreiner, M. E. (1997). Helping users think in three dimensions: Steps toward incorporating spatial cognition in user modeling. *Proceedings of the International Conference on Intelligent User Interfaces*, Orlando, FL, 113–120.

Electronic Privacy Information Center. (2001). *Filters and Freedom 2.0: Free Speech Perspectives on Internet Content Control*. Washington DC: Author.

Elliott, J., Bruckman, A., Edwards, E., & Jensen, C. (2000). Uneven Achievement in a Constructionist Learning Environment*. Proceedings of the International Conference on the Learning Sciences*, Ann Arbor, MI, 157–163.

Ellis, J. B., & Bruckman, A. S. (2001). Designing palaver tree online: Supporting social roles in a community of oral history. *Proceedings of CHI: Conference on Human Factors in Computing Systems*, Seattle, WA, 474–481.

Erickson, T. (1990). Working with interface metaphors. In B. Laurel (Ed.), *The Art of Human-Computer Interface Design* (pp. 65–73). Reading, MA: Addison Wesley Publishing Company Inc.

Ericsson, K. A., & Simon, H. (1993). *Protocol analysis: Verbal reports as data*. Cambridge, MA: MIT Press.

Gay, L. R., & Airasian, P. (2000). *Education Research: Competencies for Analysis and Application* (6th ed.). Upper Saddle River, New Jersey: Merrill.

Gilutz, S., & Nielsen, J. (2002). Usability of websites for children: 70 design guidelines based on usability studies with kids. Location: Nielsen Norman Group Report.

Goldman-Segall, R. (1996). Looking through layers: Reflecting upon digital video ethnography. *JCT: An Interdisciplinary Journal for Curriculum Studies, 13*(1), 23–29.

Guha, M. L., Druin, A., Chipman, G., Fails, J. A., Simms, S., & Farber, A. (2005). Working with young children as technology design partners. *Communications of the ACM, 48*(1), 39–42.

Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments, 4*(1), 1–44.

Guzdial, M., & Rose, K. (Eds.). (2001). *Squeak: Open Personal Computing and Multimedia*. Englewood, NJ: Prentice Hall.

Halgren, S., Fernandes, T., & Thomas, D. (1995). Amazing Animation™: Movie making for kids design briefing. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Denver, CO, 519–525.

Hanna, L., Risden, K., & Alexander, K. (1997). Guidelines for usability testing with children. *Interactions, 4*(5), 9–14.

Hickey, D. T., Kindfield, A. C. H., Horwitz, P., & Christie, M. A. (2000). Integrating instruction, assessment, and evaluation in a technology-based genetics environment: The GenScope follow-up study. *Proceedings of the International Conference of the Learning Sciences*, Ann Arbor, MI, 6–13.

Holdaway, D. (1979). *The Foundations of Literacy*. New York: Ashton Scholastic.

Hourcade, J. P. (2002). *It's too small! Implications of children's developing motor skills on graphical user interfaces* (No. CS-TR-4425). University of Maryland Computer Science Department.

Hoysniemi, J., Hamalainen, P., & Turkki, L. (2003). Using peer tutoring in evaluating the usability of a physically interactive computer game with children. *Interacting with Computers, 15*, 203–225.

Hutchinson, H., Mackay, W., Westerlund, B., Bederson, B. B., Druin, A., Plaisant, C., et al. (2003). Technology probes: Inspiring design for and with families. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Ft. Lauderdale, FL, 17–24.

Inkpen, K. (1997). Three important research agendas for educational multimedia: Learning, children and gender*. Proceedings of Educational Multimedia '97*, Calgary, AB, 521–526.

Inkpen, K. (2001). Drag-and-drop versus point-and-click: Mouse interaction styles for children. *ACM Transactions Computer-Human Interaction, 8*(1), 1–33.

Inkpen, K., Gribble, S., Booth, K. S., & Klawe, M. (1995). Give and take: Children collaborating on one computer. *Proceedings of the ACM*

Conference on Human Factors in Computing Systems, Denver, CO, 258–259.

Ishii, H., & Ullmer, B. (1997). Tangible bits: Towards seamless interfaces between people, bits and atoms. *Proceedings of CHI '97*, 234–241.

Iversen, O. S. (2002). Designing with children: The video camera as an instrument of provocation. *Proceedings of the Conference on Interaction Design and Children*, Eindhoven, Netherlands, CD-ROM.

Jensen, J. J., & Skov, M. B. (2005). A review of research methods in children's technology design. *Proceedings of the Conference on Interaction Design and Children*, Boulder, CO, 80–87.

Joiner, R., Messer, D., Light, P., & Littleton, K. (1998). It is best to point for young children: A comparison of children's pointing and dragging. *Computers in Human Behavior, 14*(3), 513–529.

Jones, T. (1992). Recognition of animated icons by elementary-aged children. *Association for Learning Technology Journal, 1*(1), 40–46.

Kail, R. (1991). Developmental Changes in speed of processing during childhood and adolescence. *Psychological Bulletin, 109*, 490–501.

Kay, A. (1972). A personal computer for children of all ages. *Proceedings of the ACM National Conference*, Boston, MA.

Kemp, J. A. M., & van Gelderen, T. (1996). Codiscovery exploration: An informal method for the iterative design of consumer products. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & I. L. McClelland (Eds.), *Usability evaluation in industry* (pp. 139–146). London: Taylor & Francis Ltd.

Kolodner, J., & Guzdial, M. (1996). Effects with and of CSCL: Tracking learning in a new paradigm. In T. Koschmann (Ed.), *CSCL: Theory and practice*. Mahwah, New Jersey: Lawrence Erlbaum Associates.

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

Lamberty, K. K., & Kolodner, J. (2005). Camera talk: Making the camera a partial participant. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Portland, OR, 839–848.

MacFarlane, S., Sim, G., & Horton, M. (2005). Assessing usability and fun in educational software. *Proceedings of the Conference on Interaction Design and Children*, Boulder, CO, 103–109.

Markopoulos, P., & Bekker, M. (2002). How to compare usability testing methods with children participants. *Proceedings of the Conference on Interaction Design and Children*, Eindhoven, Netherlands, 153–158.

Martin, F., & Resnick, M. (1993). LEGO/Logo and Electronic Bricks: Creating a scienceland for children. In D. L. Ferguson (Ed.), *Advanced Educational Technologies for Mathematics and Science* (pp. 61–90). Berlin Heidelberg: Springer-Verlag.

McElligott, J., & van Leeuwen, L. (2004). Designing sound tools and toys for blind and visually impaired children. *Proceedings of the Conference on Interaction Design and Children*, College Park, MD, 65–72.

Miller, C. (1998). Designing for kids: Infusions of life, kisses of death. *Paper presented a the Game Developers Conference*, Longbeach, CA.

Miller, L. T., & Vernon, P. A. (1997). Developmental changes in speed of information processing in young children. *Developmental Psychology, 33*(4), 549–554.

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

Nielsen, J. (1993). *Usability Engineering*. London: Academic Press.

Nix, D., Fairweather, P., & Adams, B. (1998). Speech recognition, children, and reading. *Proceedings of the SIGCHI Conference on Human Factors in Computings Systems,* Los Angeles, CA, 245–246.

O'Hare, E. A., & McTear, M. F. (1999). Speech recognition in the secondary school classroom: an exploratory study. *Computers & Education, 3*(8), 27–45.

O'Malley, C., & Fraser, D., (2004). *Literature Review in Learning with Tangible Technologies*. NESTA Futurelab Report.

Oosterholt, R., Kusano, M., & Vries, G. D. (1996). Interaction design and human factors support in the development of a personal communicator for children. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Vancouver, Canada, 450–457.

Papert, S. (1972). *On making a theorum for a child*. Paper presented at the ACM National Conference, Boston, MA.

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

Papert, S. (1987). Computer criticism vs. technocentric thinking. *Educational Researcher, 16*(1), 22–30.

Papert, S. (1992). *The Children's Machine.* New York: Basic Books.

Pea, R. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2*(2), 137–168.

Pea, R. (1987). The aims of software criticism: Reply to Professor Papert. *Educational Researcher, 16*, 4–8.

Piaget, J. (1970). *Science of Education and the Psychology of the Child*. New York: Orion Press.

Price, S., Rogers, Y., Scaife, M., Stanton, D., & Neale, H. (2003). Using 'tangibles' to promote novel forms of playful learning. *Interacting with Computers, 15*(2), 169–185.

Repenning, A., & Fahlen, L. E. (1993). Agentsheets: A tool for building domain-oriented visual programming environments. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Amsterdam, Netherlands, 142–143.

Resnick, M. (1994). *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*. Cambridge, MA: MIT Press.

Resnick, M., Bruckman, A., & Martin, F. (1996). Pianos not stereos: Creating computational construction kits. *Interactions, 3*(5), 40–50.

Resnick, M., & Rusk, N. (1996). The Computer Clubhouse: Preparing for life in a digital world. *IBM Systems Journal, 35*(3–4), 431–440.

Resnick, M., & Silverman, B. (2005). Some reflections on designing construction kits for kids. *Proceedings of the Conference on Interaction Design and Children*, Boulder, CO, 117–122.

Rubin, J. (1994). *Handbook of Usability Testing*. New York: John Wiley and Sons, Inc.

Scaife, M., & Rogers, Y. (1999). Kids as informants: Telling us what we didn't know or confirming what we knew already? In A. Druin (Ed.), *The design of children's technology* (pp. 27–50). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Scaife, M., Rogers, Y., Aldrich, F., & Davies, M. (1997). Designing for or designing with? Informant design for interactive learning environments. *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, Atlanta, GA, 343–350.

Scardamalia, M., & Bereiter, C. (1994). Computer support for knowledge-building communities. *The Journal of the Learning Sciences, 3*(3), 265–283.

Schneider, K. G. (1996). Children and information visualization technologies. *Interactions, 3*(5), 68–73.

Schneider, K. G. (1997). *The Internet Filter Assessment Project* (*TIFAP*). Retrieved March 16, 2007, from http://www.bluehighways.com/tifap/learn.htm

Schuler, D., & Namioka, A. (Eds.). (1993). *Participatory design: Principles and practices*. Hillsdale, New Jersey: Lawrence Erlbaum Associates, Publishers.

Skinner, B. F. (1968). *The technology of teaching*. New York: Appleton-Century-Crofts.

Soloway, E., Guzdial, M., & Hay, K. E. (1994). Learner-centered design: The challenge for HCI in the 21st century. *Interactions, 1*(1), 36–48.

Soloway, E., Jackson, S. L., Klein, J., Quintana, C., Reed, J., Spitulnik, J., et al. (1996). Learning theory in practice: Case studies of learner-centered design. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Vancouver, Canada, 189–196.

Songer, N. B. (1996). Exploring learning opportunities in coordinated network-enhanced classrooms: A case of kids as global scientists. *The Journal of the Learning Sciences, 5*(4), 297–327.

Stewart, J., Raybourn, E. M., Bederson, B., & Druin, A. (1998). When two hands are better than one: enhancing collaboration using single display groupware. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, Los Angeles, CA, 287–288.

Strommen, E. (1994). Children's use of mouse-based interfaces to control virtual travel. *Proceedings of the ACM Conference on Human Factors in Computing Systems: Celebrating Interdependence*, Boston, MA, 229.

Strommen, E. (1998). When the Interface is a Talking Dinosaur: Learning Across Media with ActiMates Barney. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, Los Angeles, CA, 288–295.

Strommen, E., & Alexander, K. (1999). Emotional Interfaces for Interactive Aardvarks: Designing Affect into Social Interfaces for Children. *Proceedings of the ACM Conference on Human Factors in Computing Systems: the CHI is the limit*, Pittsburgh, PA, 528–535.

Taylor, R. P. (Ed.). (1980). *The computer in the school, tutor, tool, tutee.* New York: Teachers College Press.

Thomas, J. R. (1980). Acquisition of motor skills: Information processing differences between children and adults. *Research Quarterly for Exercise and Sport, 51*(1), 158–173.

Tinker, R. (1993). *Thinking about science*. Concord, MA: The Concord Consortium.

van Kesteren, I. E., Bekker, M. M., Vermeeren, A. P. O. S., & Lloyd, P. A. (2003). Assessing usability evaluation methods on their effectiveness to elicit verbal comment from children subjects. *Proceedings of the Conference on Interaction Design and Children*, Preston, UK, 41–49.

Walker, D. F. (1987). Logo needs research: A response to Professor Papert's paper. *Educational Researcher*, 9–11.

Winograd, T. (1996). Profile: Kid Pix. In T. Winograd (Ed.), *Bringing design to software* (pp. 58–61). New York: ACM Press.