Tinkering Together: enabling synchronous creativity and distributed collaboration for kids

Agnes Chang
MAS 712: Technologies for Creative Learning
Professor Mitchel Resnick
December 10th, 2008

"An idea or product that deserves the label 'creative' arises from the synergy of many sources and not only from the mind of a single person"

(Csikszentmihalyi, 1996)

Despite technological improvements in network and increased availability of computers, the predominant paradigm in creative environments designed for children today remains that of the one-child-one-computer single interaction. The Internet allows communication with distant corner of the globe, and yet nearly all creative activity a child can do with a computer remains limited to the single-user. At the same time, with society's increasing tendency to migrate out to suburbs, children are increasingly isolated in the physical world as well. This unfortunately derives a child of innumerable learning opportunities, not the least of which is that creative work necessarily arises from the synergy of many sources. In the same manner as paintbrushes and LEGO bricks lead a child to think in different and creative ways, likewise the computer today is the perfect host of a truly empowering digital medium that challenges children not only to create an unlimited variety of objects, but also to create in infinitely different *ways*.

In this paper, I would like to propose a set of design guidelines for a digital medium that takes a decentralized, low-floor-high-ceiling-wide-walls, constructionist approach to enabling synchronous collaborative creativity. I begin by reviewing the benefits of collaborative learning, particularly when it is applied to creative work, and the requirements for enabling social learning. Next, I explain the necessity of a decentralized approach to enabling children's existing collaboration methodologies, and review existing approaches to synchronous collaboration both suited and unsuited to facilitating children's creativity. I then identify specific design challenges of synchronous collaborative software systems and list guidelines for designing the user interface. Based on these analyses, I propose a design for the Scratch platform developed at the MIT Media Lab, as well as a corresponding use scenario demonstrating how the design and use addresses the requirements of each of the three levels previously mentioned: social, programmatic, and interface. Finally I discuss implications for further work in designing environments to support synchronous collaborative creativity for children.

Social Learning: Transcending Distance

There exist numerous theories on roles of technology that support social learning, by which I mean learning through interaction with peers. Amongst them include leveraging distributed intelligence, allowing personal reflection though interaction with different people of different backgrounds, acting as a boundary object across different communities (Fischer, 2007). There has also been much attention to contemporary society's need to regain social capital, in the sense of mutual reciprocity, the resolution of dilemmas of collective action, and the broadening of social identities (Putnam, 1995). Through interacting with peers, children also learn the skills of communication, negotiation, critical thinking, organization, etc. that are critical to being an adult citizen in society. In short, much can be gained from interaction with people from outside of one's immediate physical space and immediate community.

In addition, collaboration and interaction with others opens access to inspiring ideas and empowering skills that leads to new dimensions of thinking and creativity. All of this would be even more effective if students had the opportunity to interact with each other from across distant communities and participate in learning activities with each other. In today's Internet age, there are no longer technological limitations, and yet the power of technology is still unharnessed. Today's Internet affords the largest source of information and knowledge in the world, enabling a new demand-pull approach to learning (Brown, 2008).

Instead of the traditional supply-push "spoon feeding" model of learning the Internet now enables passion-based learning, where the student, motivated by either wanting to become a part of a particular community or just want to learn how to perform something, can find the resources to do so; except, the most precious resource of all, people, can only be reached in limited ways. Children in society today need a digital medium which affords:

- intimate interaction with people across distances physical or social
- actively learning from, and sharing knowledge with, other people
- self-organization, self-mediation, and freedom of decision about creative content
- easy linkage and integration with the rich multimedia resources of the Internet

Defining Collaboration: Methodologies and Paintbrushes

To investigate how to design a digital medium that leverages the unique affordances of the Internet to promote collaboration, we must first clarify the specific type of collaboration we seek to address, and identify examples of successful traditional media that enable this type of collaboration to use as our guide. Collaboration could fall on a spectrum along two dimensions of interactivity between peers and synchronicity (Figure. 1). Much work has already been achieved in asynchronous collaboration on creative work, ranging from posting of critiques messages in Internet forum (e.g. http://forums.cgsociety.org/), posting resources to an open-access repository (e.g. http://resources.scratchr.org/), in creating interactive objects individually to participate in a virtual world (Bruckman, 1997), but what is truly missing in the picture is the ability to interact through a digital medium and co-create an expressive work in real-time.

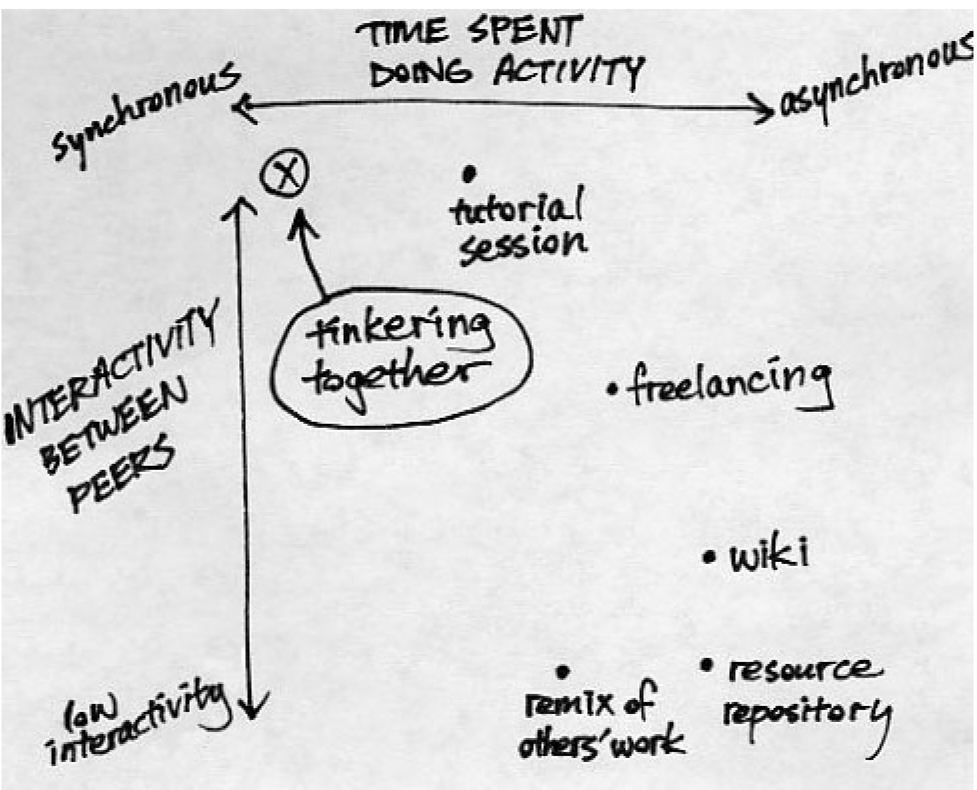

Figure. 1. Different types of collaboration, mapped to the two axes of interactivity and synchronicity.

Synchronous collaboration is a unique and special task that has many requirements. To be truly successful, the digital medium must allow the full range of possible interactions: including observation, resource-sharing, knowledge-sharing, coordinated action, problem-solving, decision-making, etc.

We take as our models two successful external play objects for kids that are established creative media: paintbrushes and LEGO bricks. These two objects have not only passed the test of time and continues to be two of the most popular creative playthings for children, but as testament to their successful design, are a creative medium that continues to be chosen by people of all ages. Both the paintbrush and LEGO bricks possess the qualities of "low floor, high ceiling, and wide walls" (Papert, 1980). This refers to the fact that both media are easy to pick up, allowing everyone to participate; they can be used to make highly technically-challenging works, be it professional painting or a life-size LEGO submarine model; and they enable a variety of uses they might not have necessarily been designed for.

When talking about supporting collaboration with computers, conferencing systems and shared whiteboard applications quickly come to mind. However, these do not enable the full range of activity. Many restrict turn-taking to maintain group order and to ensure a certain level of awareness amongst participants of each others' likely activities (Dourish 92). Certainly this is detrimental to creativity. The birth of many of the most creative ideas have been individuals jumping out of turn or taking on a perspective outside of their usual role. In the case of a group of children sharing a tub of LEGO bricks, or a set of paints, talking to each other and signaling their intentions as they work, we can see that instead of centralized restrictions, group moderation can be achieved by taking a decentralized approach. Rather than limiting users' actions, creative digital media should be thorough in supporting the collaborators' self-mediation through social negotiation. Other examples of technology support for creativity take the approach of developing different functionalities (in essence restricting some functionalities) in cases of collaborative work in the adult world (Repenning, 1995). In these cases there are specific deliverables for the collaborative activity, and participators have specific roles. Likewise this is restrictive, and it also unnecessarily enforces a strict hierarchy amongst participators, both of which are undesirable for children's creative work. In the cases where role designation is necessary among a group of children, the authority should by socially determined and agreed upon by the participants. In addition to this being a more sustainable way of ensuring order and satisfaction, it also gives children the chance to acquire the skills that will make them a citizen of society.

Also, we see that given a healthy group dynamics (and I address this issue towards the end of this paper), children are perfectly capable of longer-term self-organizing and self-mediation (Bruckman, 2007). Amongst the many evidence of successful self-organization of kids can be found in the Scratch community (Aragon, 2008). One example is a group of 15 participants, self-titled the “Green Bear Group” that was founded by three children, ages 8, 13 and 15. The group members pooled together their skills to create video games by first voting on which games to develop, and then each member has a specific skill, such as art, music, programming, or storytelling, which they contribute to the game by downloading an unfinished version, editing the program, and re-uploading a new version. "The active participants in Green Bear Group have, for the most part, never met one another, live in different time zones, and do not even know each other’s real names." (ibid.)

Thus, I summarize the implications for design:

- maximize the medium's affordance for each individual’s creative expression and active participation, while minimizing the threshold of entry to become a participator
- maintain overall quality by enforcing a minimal set of community standards and allowing users high degrees of autonomy for discussion, input, and negotiation

- children are more intelligent and more creative than is often assumed, and can achieve great things if given a supportive environment to do so

Design Guidelines: User Interface

There exist a number of challenges which are important for all forms of collaboration but absolutely fundamental for a decentralized and low-floor-high-ceiling-wide-walls approach collaboration. These include questions such as:

- how to ensure that users do not step over each other's toes while working?
- how ensure users to feel comfortable creating personal work under public scrutiny?
- how to enable effective communication with to peers the user cannot see or does not even necessarily know?
- with new and additional models of working, how to enable the user(s) to transition seamlessly amongst them?

These issues are intertwined and heavily based in trust, since a prerequisite to working comfortably is the knowledge that your peers will not interfere with your work without your permission. Communication is also necessarily the source of awareness, which in turn mediates social control. However, for ease of discussion, here I address them as four challenges: territoriality, communication, control/awareness, and transition.

Humans are fundamentally spatial animals, but in general the use of space is subconscious. Thus the designer must study behavioral patterns, but ultimately the spaces they design must allow people to determine the usage protocol for themselves. For the case of synchronous collaboration, many studies of the usage patterns of physical spaces can be applied to people's use of digital spaces. Studies of workspaces have revealed that people feel comfortable creating when they have a sense of *place*, which does not necessarily have to be directly mapped to the characteristics (such as size) of the *space* (Harrison, 1996). Also, an empirical study on the use of traditional shared table-top workspaces shows that usage patterns cleared indicated that there are three types of table-top territories: personal, group, and storage (Scott, 2003). These spaces were not explicitly designated, but collaborators felt more comfortable having the choice to first privately sketch out initial ideas, and perhaps test them, before presenting them to the shared "group" space for evaluation and discussion.

These studies also noted that in traditional tabletop collaboration, ownership of the tabletop objects was usually mitigated through social protocol and not physical restrictions, in the same way that children share a tub of LEGO bricks. However, many of these subtle turn-taking and permission-granting signals in a face-to-face scenario are lost in the transfer to a virtual workspaces. Thus it is necessary to indicate as much as possible, through passive displays, the activity and intents of the users, in order to enable users to coordinate their actions effectively. For this we can take inspiration from other online synchronous communication tools, such as AOL's instant messaging client. Users can describe their current status, but also the client program automatically detects when a user has been away from the computer, or is currently typing in the chat window, or is busy with another application on the computer.

In the case of a programming environment, another similar programmable virtual environment for children (Jackman, 2006) that implemented a collaborative version identified three distinct spaces within the environment: the student-shared code space, the global run-time space, and the mentor-space in which teachers could assist students. The spatial divisions in the case of a creative digital medium intended for use outside of a school context might be different, but also possess separate spaces within the collaboration methodologies, each of which will require support from the software in order to ensure a smooth work flow. These include enabling the sharing of objects, transitioning between activities, between personal and group work, and between the collaboration and external work (e.g. pulling a reference book from the bookshelf next to the computer).

Thus we propose the following guidelines for a collaborative creative user interface taking a low-floor-high-ceiling-wide-walls approach:

- enable both public and private spaces for work but being aware that overuse of private spaces can hinder group transparency and interpersonal interaction
- show environmental indicators of the activity of all users at all times in a way that would enable trust and comfort, and not interfere with the creative task
- enable all forms of communication, and support communication about the work. Try to reproduce digital forms of real-world communication channels such as body language
- allow seamless transition between all manner of work spaces, coding, executing, etc., and including to workspaces and resources external to the environment

Design Proposal: A Collaborative Scratch

Scratch (http://scratch.mit.edu), often referred to as a new programming language for kids, was developed by the Lifelong Kindergarten Research Group at the MIT Media Lab. Scratch enables children to create not only games but also interactive stories, animations, music, and art (Resnick, 2003). Scratch takes the approach of "tinkerability", referring to the ease by which children could playfully put together fragments of computer programs, try them out, take them apart, and recombine. The choice to apply the guidelines outlined above to the Scratch platform was based on a number of reasons:

First, the design of Scratch is firmly based upon the constructivist approach, such that it comprises a solid set of well-designed and well-used tools which already fulfill the principles of low-floor-high-ceiling-wide-walls, at least in the single-user scenario.

Secondly, Scratch already has a dedicated and enthusiastic user base (referred to as Scratchrs), many of whom, especially experienced users, are actively interested in discussing the design of new functionalities.

Last but not least, Scratch is not simply a programming language -- the word Scratch equally refers to the web community, where Scratchrs already engage in significant levels of collaboration such as knowledge sharing, "freelancing", remixing, and organization. These are indicators that the community is ready for synchronous collaboration.

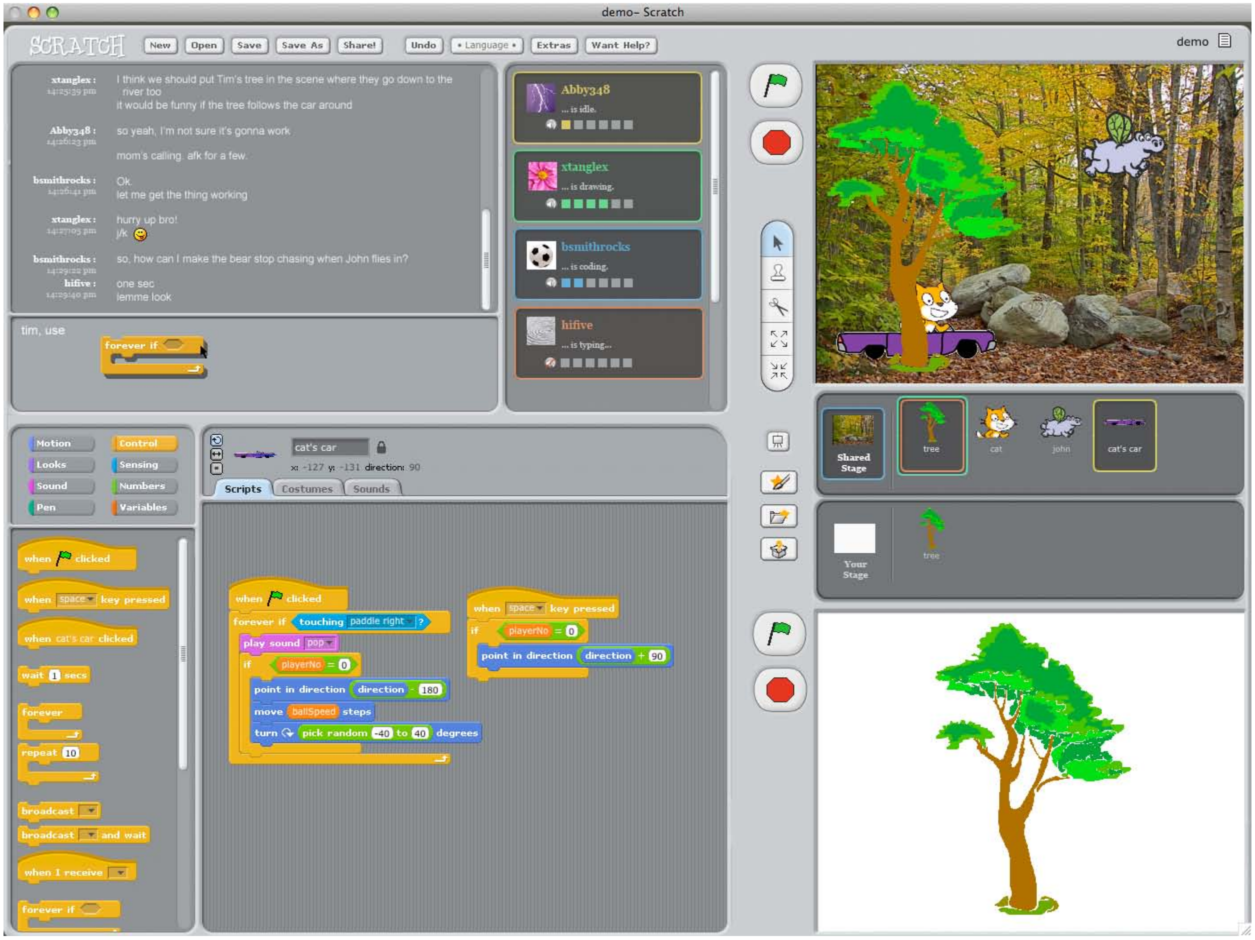

Figure. 2. Full screenshot of the user interface of the proposed collaborative version of Scratch.

The design screenshot (Figure.2) shows all three sections of the proposed user interface. The communication interface is located on the top left, the code/scripting area on the bottom left, and the canvas, or "stage" on the right-hand side of the screen. Below I describe the design of each section in detail.

Figure. 3. Detail of the proposed chat interface.

To enable multiple channels of communication, and to consider possible equipment limitations, both a voice-chat and text-chat functionality were added (Figure.3). The text-chat functionalities are more or less consistent with common instant message protocols. To the left is the chat transcript, showing user names, message time, message content, as well as expressive elements such as emoticons. An additional feature unique to the environment is the ability to also drag blocks of code into the chat box, and literally "speak code". The intention is to enable users to discuss code more effectively by being able to accurately refer to specific blocks, or even segments of code that comprise multiple blocks.

On the right is the user representation space, or the presence box, in contrast to the chat box. Here displays is a small image as the user's avatar, the user's name, and color assignment which is used universally in the workspace to indicate the user's actions whenever possible. Underneath the user name is a status message which will automatically show which part of the many workspaces (chat, coding, stage, paint, external, etc.) the user is working in. The colored blocks underneath the user name indicate voice levels and availability of voice chat. Thus, if a user does not have audio output on the computer they are working on, they will be aware that voice communication is going on, even if they cannot hear it. Each of these indicators are meant to increase users' awareness of each other and thus promote group cohesion.

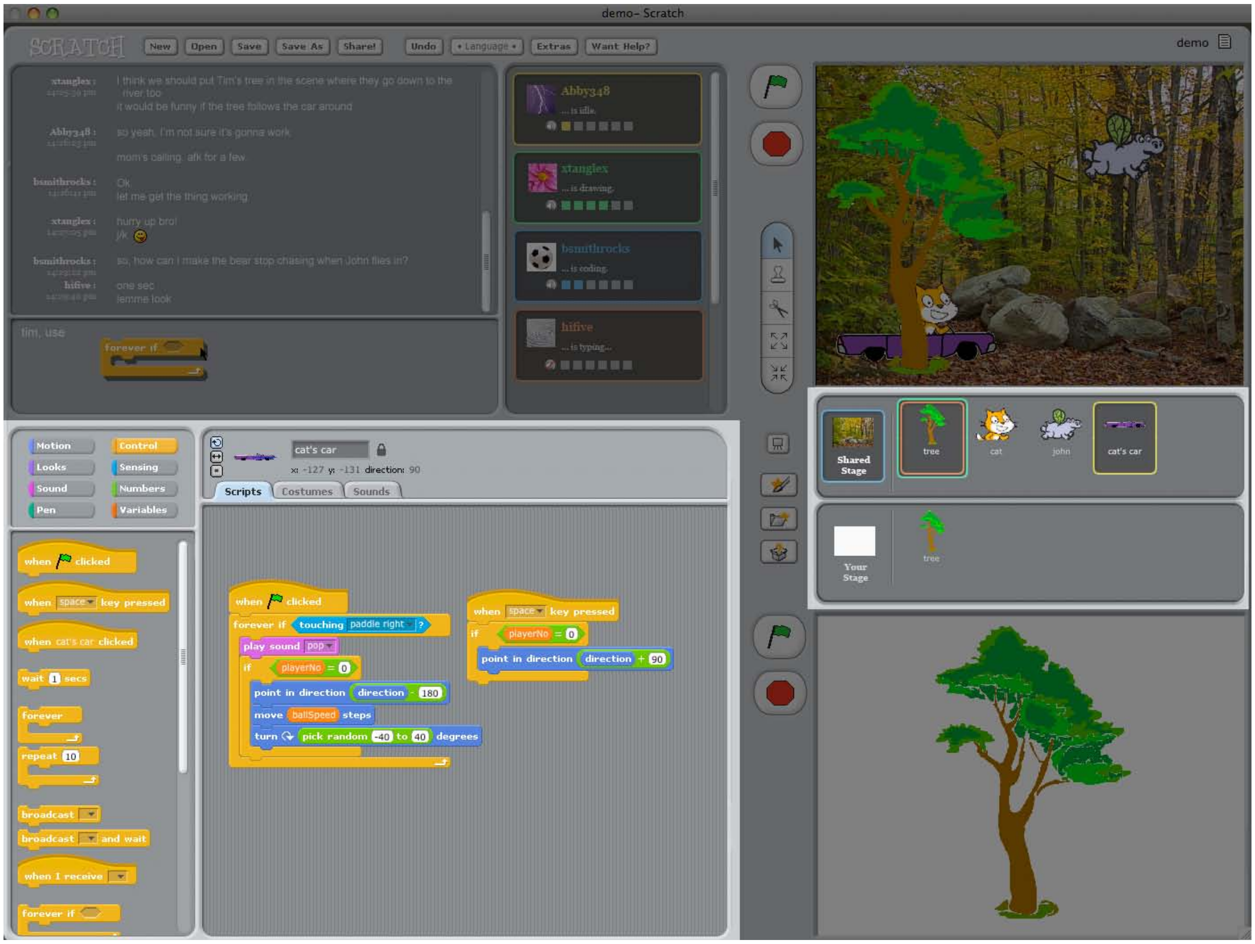

Figure. 4. The script (code) workspace.

Additional indicators of activity are located in the various workspaces (Figure.4), in this case showing direct mapping of user activity and space, rather than a remote indicator describing the space. To the right, the sprites in the library are highlighted with a colored halo according to the color of the user currently using the sprite. As the middle sprite example shows, multiple users can access one sprite at the same time.

As a sprite is selected (and one user can only select one at one time), the code accompanying the sprite will display in the user's code space. There is only one code space, but the user can switch between any of the sprites by simply selecting them. To ensure that users will see the same space, if the space belongs to the group, code layouts will be synchronized across different users. That is, if two different users look at the same sprite, they will see exactly the same code space except for camera characteristics such as pan and zoom.

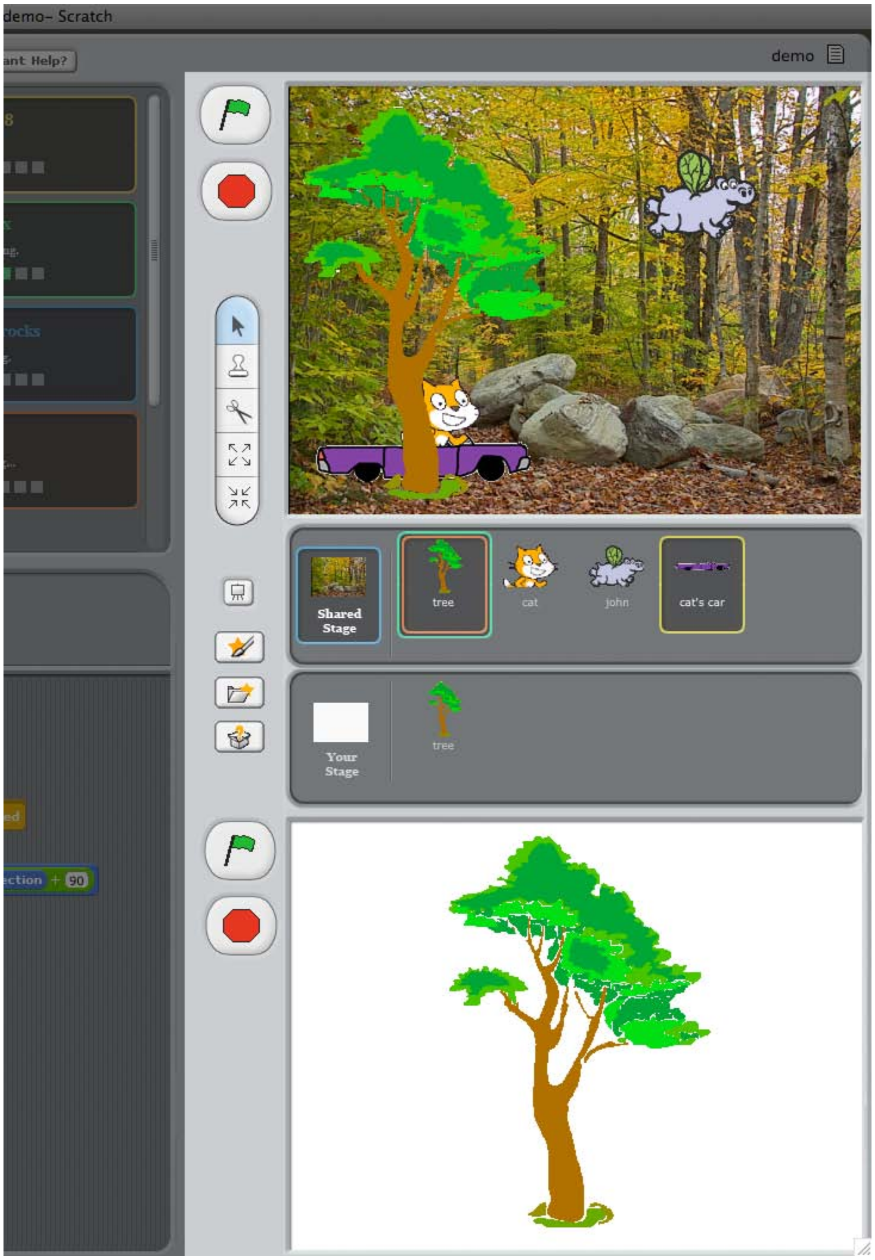

Figure. 5. The dual private-and-public stage workspaces.

Finally (Figure.5) shows the dual private-and-public workspaces. This enables an individual to be able to create sprites in his/her private workspace, test it out, and revise it before added it to the group project, which exists in the group space. Drag-and-drop as well as copy-and-paste functionalities are valid both across private and shared stages, as well as the libraries, the code space, and the chat interface.

Use Scenarios: A Session of Elm Tree Production Company

Here I imagine the scenario of a "company" within the Scratch community, comprised of four friends who have arranged to meet together at a fixed time to make a narrative project through the collaborative Scratch environment.

Sally and Henry Johnson are siblings. Henry is the older one. Henry used to go to school with Eric, so that's how Henry and Sally know him, but they've never met Eric's friend Bob, although they know him on the Scratch website. Eric met Bob through summer camp, so Bob actually lives across the country from the other three members of the group. They join up in the collaborative Scratch environment, and they wait until they see in the presence box that everyone connected successfully. Next they test out the voice chat and confirm that everyone can both speak and hear each other. They are all set to start working.

The reason for today's get-together was that two days ago they had chatted over IM and all gotten really excited about making a story that incorporated Harry Potter-style magic and Transformers. They decided upon a basic storyline at that point and decided to work together today.

As a first scenario, we can imagine that, as with any other field that has a culture and also enables simultaneous participation, situated learning (Brown, 1989) and a mentor-apprenticeship model of interaction is now possible. The voice chat, and ability to share the paint editor of any global sprite enables a novice coder to observe the creative process of a more experienced Scratchr, and the private space of a canvas enables a novice to feel comfortable testing things out before bringing them to the table.

Sally is a newcomer to Scratch who has made a few basic projects of her own, but she feels inexperienced in drawing sprites of her own. Eric had told her earlier that Bob is really good at drawing (Bob's projects have been features on the front page of the web site a number of times) and indicated that Bob is willing to show her how he does his work. After everyone is connected, Bob goes ahead and drops the default Scratch cat sprite on the shared stage so Sally can see it. He tells Sally to look at the cat's first costume. Sally also clicks on the Scratch cat and opens up the appropriate costume. Here she see the pencil tool, controlled by Bob (as indicated by the appropriately-colored halo) and she watches at he edits the cat. The two of them agree to give the cat a car, so Bob shows how to draw a car on the cat. She asks him questions about using different pencil sizes through voice-chat as he is working.

After the car demonstration, Bob asks Sally what she wants to draw and offers to guide her through it. Since for the narrative Sally had decided that the main character would be a flying hippopotamus fairy, she wanted to work on that. Sally doesn't really feel like she wants Bob to watch her though, so she arranges with him to let her work on it on her own stage while the boys make the background for the story, and then they can change it together once she drags it from her local sprite library to the shared sprite library.

In another scenario, the shared chat interface and shared coding environment enables discussion of coding and knowledge sharing.

Later on, Bob is stuck on coding the Mr. Bear sprite on the shared stage, trying to get it to jump over the rubble sprite. He asks his question, and Henry says that he could use position-x and -y blocks. However, Henry was talking too fast and the voice chat sounded fuzzy, so Bob didn't understand. He asks Henry to clarify. Henry says, "I'll put the ones I'm talking about in the chat." In everyone's chat transcript, soon enough a message says "Henry: 2:34 PM: [block] [block]". Bob drags those two blocks from the chat transcript down to his code space, effectively duplicating them, and inserts it in what he thinks are the appropriate places in his code. He then tries to execute his code segment, but it turns out that Mr. Bear falls off the stage and disappears instead. Once again he calls for help and this time Eric comes to the rescue (Henry started coding the stage background swap function and thus was slower to respond). Eric pulls up the very same Mr. Bear sprite by selecting it in the shared sprite library. He makes the Bear reappear but using some position blocks, and then he looks at Bob's code. He finds the issue: "You just accidentally put the x-position and the y-position backwards, that's all," he tells Bob.

At other phases of collaboration where users must work closely together to solve a greater problem, the environment enables responsive, communicative discussion, as well as a shared workspace.

Henry and Eric have started working on the hardest part of the code. At this point in the story they wanted to insert a little mini-game, so the audience must to control the hippopotamus, and dodge a number of fireballs to get to the next stage of the story. Problem is, the way the fireball falls will depend on both the number the hippo has dodged at the time, as well as how the audience scored in another mini-game earlier in the narrative. They hadn't really figured out how they would do it, but they decided that it will be faster if they each take one half of the job, so Henry will try to calculate how the fireball will fall given the number the hippo has dodged, while Eric will incorporate the audience's score in the the fireball path. They both open the fireball sprite on the global stage and start a attaching a group of blocks. They consult each other on the integration of the fireball's path and speed and location as they work. Eventually, however, they find that their path and speed scripts work independently, but doesn't work the way they want it to when they combine the scripts, which is what they ultimately need to do.

Eric decides to try out something first, "Ok, hang on. Leave that there so we don't lose it and let me try something on my stage first." He drags the shared fireball sprite down to his own stage, and Henry can see in his presence box that Eric is now active in his local stage. After a slight pause Henry hears from him, "Ah, okay I get it now. I should use the distance block instead." Soon his colored halo marker is back on the same fireball sprite as Henry. Henry watches as Eric rearranges the code, and voila, it does indeed work.

Finally, the collaborative environment would also enable users to write scripts on the fly, and create interactive objects that interact with the other players' creations. This type of play prompts quick thinking, quick problem-solving, and rewards creative and fun solutions.

After the Johnsons have left, the file has been uploaded, and the job is done, Bob and Eric decide to have a little fun.

"Mr. Bear is going to eat the flying hippo~", Eric taunts. He very quickly draws a new costume for Mr. Bear with vampire teeth, and adjusts the scripts to that the Mr. Bear sprite is now in an infinite loop, stuck chasing the hippopotamus sprite. Eric starts dragging the hippopotamus sprite all around the stage, and Bob comes to the hippo's rescue. He imports a rocket launcher sprite he had made for an earlier project, and adjusts the hippo's script so that whenever Mr. Bear gets too close to him, the rocket launcher will change costume to show a trail of flame in the direction of Mr. Bear...

Future Work: Affordances and the Ecosystem

There are still many directions regarding functionality design which are unexplored here. The first direction concerns further developing the internal affordances of the environment. The second direction looks outward to the ecosystem in which such a creative platform might be implemented, with attention to integrating it with the rest of the Scratch ecosystem.

One limitation of the affordances of the current design is that the usage of tools within the environment remains unchanged from the single-user paradigm. When paintbrushes in the physical world are mixed together, the additional variable of another person and another paintbrush yields unexpected possibilities. In the digital medium, analogous new blocks or new special multi-user effects would encourage collaborative experimentation and most likely enhance the collaboration experience. The current design guidelines also do not account for different individual learning styles. For example, if one considers the two classic learning styles of tinkerer versus planner, planners might wish for a way to take notes for creative decisions made jointly by the group. Extending the use scenario, this could take the form of a shared note-taking space where a list of scenes and quick descriptions could remind everyone of the decided order of a narrative project.

Additionally, a few chances to observe children actually at work in Scratch proves that group dynamics strongly affect the nature of collaboration. As previous studies (Dillenbourg, 1996; Benford, 2000) have noted, collaboration is a social structure, and while undeniably it can be a positive experience, in some cases it can have a negative effect. The group dynamics, including factors such as dominance and mediation, might be the single-most influencing factor on the interactions, the collaboration outcome, and the collaboration experience. It remains to be determined to what degree the design of a digital medium or a user interface could affect group dynamics, be it directly or indirectly through design decisions regarding functionalities, but it should be recognized that while the design proposed here was derived from analyses of existing collaborative paradigms, the nature of social interactions also depend on the resources available to mediate the exchange (Crook, 1998). Thus, iterative design cycles are necessary. An understanding of the effects of group dynamics would probably yield better results during the evaluation phase. Finally, comparison of creative collaboration with existing models of software development (such as extreme programming) might also yield insight into the interaction paradigms at play.

The other future direction of expansion addresses the fact that Scratch really refers to the entire ecosystem of Scratch platform, Scratch board, website, and community. Each intersection of the new collaborative Scratch and each of the existing elements will require investigation and design. To begin with, there will need to be an online "space" within which Scratchrs can "meet up" prior to joining together in a workspace. To fully implement the original intentions of enabling Scratch to act as a boundary objects and connect people from different communities, or supporting long-tail learning and discovery of similar interests in the community (Brown, 1998) would first require a high-level of trust within Scratch community members. In turn. an establishment of trust would first require better structural support for socialization on the website.

Next, the use of collaboration tools requires experience and previous studies have shown that the way features work need to make more explicit, e.g. that tools should give more indications to users when possibility for collaborative functionalities existed (Benford, 2000). Also, a method of transition between single-user and multi-user mode requires careful designed, taking into consideration all the details of when users might seek one or the other work environment. The final challenge is the method of keeping records (Schneiderman, 2007). Rich-history keeping aligns with the goals of allowing children to learn from the actions of others; however, this material can easily become overwhelming in volume, or so disorganized as to be useless. Thus, careful thought should be applied to the use of the recorded material, the decisions of what to record, how recorded materials could be accessed. A few use scenarios of how persistent interactions can be beneficial to collaborative creativity would probably be meaningful.

Conclusion

As we are nearing the end of the Web 2.0 era, and innovative thinkers have turned their attention to imagining the future Web 3.0, the interaction between child and computer should not be restricted to a limited dyadic loop between one child and one computer. As many other aspects of our work are now integrated with the Internet, designers of learning technologies for kids should take on the challenge of designing a creative medium that encourages new types of collaboration and new types of learning.

In this paper I described a constructivist approach to enabling synchronous collaboration to support kids' creative work. I describe the three layers of requirements: social, programmatic, and interface which all need to be addressed in order to design an effective a digital medium that supports synchronous collaborative creativity.

References

Aragon, C. et. al. (2008). A Tale of Two Online Communities - Fostering Collaboration and Creativity in Scientists and Children. Submission to SIGCHI conference on Human factors in computing systems 2009.

Benford, S. et. al. (2000). Designing storytelling technologies to encouraging collaboration between young children. Proceedings of the SIGCHI conference on Human factors in computing systems 2000. pp. 556-563

Brown, J. S. and Duguid, P. (1989). Situated Cognition and the Culture of Learning. Educational Researcher, Jan-Feb 1989, vol. 18, no. 1, pp. 32-42.

Brown, J. S. and Adler, R. P. (1998). Minds on Fire: Open Education, the Long Tail, and Learning 2.0. EDUCAUSE Review, vol. 43, no. 1 (January/February 2008). pp 15-32.

Bruckman, Amy S. MOOSE Crossing: Construction Community and Learning in a Networked Virtual World for Kids. Ph.D. Thesis, MIT Media Laboratory. June, 1997.

Crook, C. (1998). Children as computer users: the case of collaborative learning. Computers & Education, vol. 30, iss. 3-4, 5 April 1998, pp. 237-247.

Csikszentmihalyi, M. (1996). Creativity — Flow and the Psychology of Discovery and Invention. New York, NY: HarperCollins Publishers.

Dillenbourg, P. et. al. (1996). The evolution of research on collaborative learning. In E. Spada & P. Reiman (Eds) Learning in Humans and Machine: Towards an interdisciplinary learning science. Oxford: Elsevier. pp. 189-211.

Dourish, P. and Bellotti, V. (1992). Awareness and coordination in shared workspaces. Proceedings of the 1992 ACM conference on Computer-supported cooperative work. pp. 107-114.

Fischer, G. (2007). "Designing Socio-Technical Environments in Support of Meta-Design and Social Creativity", Proceedings of the Conference on Computer Supported Collaborative Learning (CSCL '2007). pp. 1-10.

Harrison, S. and Dourish, P. (1996). Re-Place-ing Space: The Roles of Place and Space in Collaborative Systems. Proceedings of the 1996 ACM conference on Computer supported collaborative work.

Inkpen, K. M. et. al. (1999). This is fun! we're all best friends and we're all playing: supporting children's synchronous collaboration. Proceedings of the 1999 conference on Computer support for collaborative learning, 1999. Article No. 31.

Jackman III, John D. A Networked Development and Runtime Environment for StarLogo TNG. M. Eng. Thesis, MIT Dept. of Electrical Engineering and Computer Science. June, 2007.

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

Books.

Putnam, R. D. (1995). Bowling Alone: America's Declining Social Capital. Journal of Democracy 6.1, 1995. pp. 65-78.

Repenning, A. and Sumner, T. (1995). Agentsheets: A Medium for Creating Domain-Oriented Visual Languages. IEEE Computer, 1995.

Resnick, M. et. al. (2003). A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantaged Communities. Proposal to the National Science Foundation.

Schneiderman, B. (2007). Creativity support tools: accelerating discovery and innovation. Communications of the ACM. Volume 50, Number 12 (2007), pp 20-32.

Scott, S.D. et al. (2004). Territoriality in collaborative tabletop workspace. Proceedings of the 2004 ACM conference on Computer supported collaborative work. pp. 294-303.