PROJECT DESCRIPTION

Major: Scratch 2.0: Cultivating Creativity and Collaboration in the Cloud

1. Introduction

In 2003, our research group at the MIT Media Lab was awarded a four-year grant from the National Science Foundation (ITR-0325828) to develop a new programming platform, called Scratch, that enables young people to create their own interactive stories, games, animations – and share their creations with one another online. The Scratch website (http://scratch.mit.edu), launched in May 2007, has become a vibrant online community, with more than 350,000 registered members sharing, discussing, and remixing one another's Scratch projects. Each day, members of the Scratch community (mostly ages 8 to 15) upload nearly 1500 new Scratch projects to the website – on average, a new project almost every minute. Overall, community members have contributed more than 500,000 projects to the website.

As young people program and share Scratch projects, they begin to develop as *computational thinkers*: they learn core computational concepts, while also learning important strategies for designing, problem solving, and collaborating (Wing, 2006, 2008). At the same time, they begin to see themselves as *computational creators*, capable and confident to design and create with computational media, not just interact with it.

As the Scratch community has grown, we have been surprised and pleased by:

- *The creativity and diversity of projects*. Scratch community members have created and shared many different genres of projects, including interactive newsletters, science simulations, public service announcements, virtual tours, animated dance contests, interactive tutorials, and many others.
- *The broad range of participants*. Although we designed Scratch for ages 8 to 15, people over the age of 15 have contributed approximately a quarter of all projects on the website. Scratch has been integrated into computer science courses not only in secondary schools, but also at several universities (including Harvard and Berkeley).
- *The high level of collaboration on the website*. Members of the Scratch community are continually remixing one another's projects, enhancing existing projects by adding new images, sounds, and programming scripts. In addition, some Scratch community members have formed self-proclaimed "companies," bringing together young people from around the world to collaborate on the design and development of Scratch projects.

Although we are excited by the creative collaborations already taking place in the Scratch community, we believe that the community is still in the very early stages of development. Our preliminary studies of activity patterns in the Scratch community have sparked many new ideas on how we could better support creativity and collaboration. We have also received thousands of feature requests from enthusiastic Scratchers, with many suggestions on how to enhance collaboration in the community. In particular, we have found that the separation of the Scratch programming application (running locally) and the Scratch online community (on the web) presents a significant obstacle to certain types of sharing, interaction, and collaboration.

In response, we propose to develop a new generation of Scratch, called Scratch 2.0, that will migrate Scratch to the "cloud," making it possible for people to author Scratch programs directly on the web. With Scratch 2.0, the process of creating a Scratch project will be analogous to creating a collaborative document using Google Doc, instead creating a file on your local machine and then uploading the file to the web. We believe that this new version will greatly enhance opportunities for creative collaboration in the Scratch community – and, more broadly, it will enable us to use Scratch as a testbed for exploring and studying how cloud-based programming can support new forms of creativity and collaboration.

We divide our research into two strands:

- ***Technological infrastructure for creative collaboration***. With Scratch 2.0, people will be able to design and program new types of web-based interactions and services. For example, they will be able to program interactions with social-media websites (such as Facebook), create visualizations with online data, and program their own collaborative applications.
- ***Design experiments for creative collaboration***. As we develop Scratch 2.0, we will run online experiments to study how our design decisions influence the ways in which people collaborate on creative projects, as well as their attitudes towards collaboration.

This project will require integration of ideas from a range of disciplines, including computer science, education, design, psychology, and sociology. We will focus especially on two questions from the NSF Program Solicitation:

- Will the research lead to the development of new technologies to support human creativity?
- Will the research lead to innovative educational approaches in computer science, science, or engineering that reward creativity?

2. Conceptual Frameworks

2.1 Social Creativity

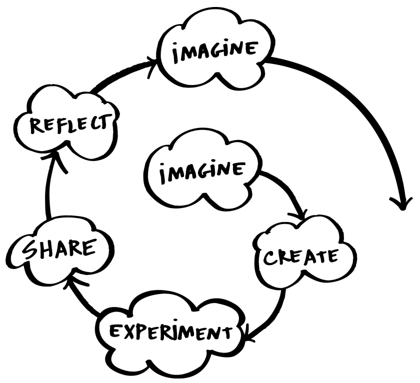
In recent decades, there has been a growing recognition of the importance of collaboration and social interaction in the creative process. Mihalyi Csikszentmihalyi, a leading proponent of this perspective, explains: "An idea or product that deserves the label 'creative' arises from the synergy of many sources and not only from the mind of a single person. It is easier to enhance creativity by changing conditions in the environment than by trying to make people think more creatively" (Csikszentmihalyi, 1996). Keith Sawyer, a member of our Advisory Board, has studied creativity in a wide variety of contexts, from corporate brainstorming sessions to improvisational jazz ensembles to kindergarten classrooms. He argues that collaboration is the key to creativity in all contexts: "We're drawn to the image of the lone genius whose mystical moment of insight changes the world. But the lone genius is a myth; instead it's a group genius that generates breakthrough innovation. When we collaborate, creativity unfolds across people" (Sawyer, 2007).

The Internet and related communications technologies open new possibilities for creative collaboration, and there are many well-publicized success stories, such as the collaborative development of Linux and Wikipedia (e.g., Raymond & Young, 2001; Tapscott & Williams, 2006). But, of course, communications technologies on their own do not guarantee collaboration.

A key challenge is how to design environments and social infrastructures that explicitly encourage and support collaboration. Yochai Benkler, a member of our Advisory Board, has emerged as a leading theorist on this issue (Benkler, 2006). Benkler has proposed a set of "design levers" that influence the level and nature of cooperation within a system (Benkler, in press). For example, Benkler points to "transparency" as an important design lever – that is, systems that are designed to enable participants to see what others are doing are more likely to foster collaborative activity. We will use the design-lever framework to guide our design of Scratch 2.0, in order to support and study new forms of collaboration within the Scratch community, as described in sections 4 and 5 below.

2.2 Design-Based Learning

In designing Scratch 2.0, we are especially interested in supporting a specific type of collaboration: collaboration on design activities. In previous research, we have found that design-based activities, such as creating interactive stories and games, offer a particularly effective way for youth with diverse interests to become engaged in exploring computational ideas (e.g., Resnick, 2002; Rusk, Resnick, Berg, & Pezalla-Granlund, 2008; Rusk, Resnick, & Cooke, 2009). As youth work on design projects, computational ideas are situated within meaningful activities, rather than presented as decontextualized concepts, as happens too often in computer science classes. We have also found that design-based approaches support the development of creative thinking (Resnick, 2006). In short, providing learners with opportunities to *create* (along with opportunities to *collaborate*) is the best way to help learners develop as creative thinkers.



In communicating our design-based approach to educators, we often describe the process in terms of a diagram that we call the *creative learning spiral*. In this process, learners *imagine* what they want to do, *create* a project based on their ideas, *experiment* with their creations, *share* their ideas and creations with others, *reflect* on their experiences – all of which leads them to *imagine* new ideas and new projects (Resnick, 2007a, 2007b). In developing Scratch and the Scratch website, we focused explicitly on how to support all phases of this learning spiral. People *create* Scratch programs by snapping together graphical programming blocks. By avoiding the obscure syntax of traditional programming languages, Scratch frees people to tinker and *experiment*, constantly trying out new possibilities. When people *share* their projects on the Scratch website, they receive comments and feedback from other members of the Scratch community, leading them to *reflect* on their experiences. As people try out other projects on the website, they *imagine* ways that they could remix and extend them.

This creative learning spiral will guide all of our efforts related to Scratch 2.0, as we aim to develop technical and social infrastructures to support learners in imagining, creating, experimenting, sharing, and reflecting.

2.3 Broadening Participation

There has been a well-documented drop in the number of students taking computer-science courses, and a very low level of participation by women and members of minority groups

(Margolis & Fischer, 2002; Margolis, 2008; Klawe, Whitney, & Simard, 2009). Although participation in the *use* of computers has improved across the population, only a narrow slice of the population is actively involved in designing and creating with computation.

There is preliminary evidence that Scratch can help attract a broader range of students to computer-science courses, for at least three reasons:

- The Scratch graphical-programming approach allows students to focus on core computational ideas, without having to worry about obscure syntax.
- Scratch connects with student interests, allowing them to work on projects that they find personally meaningful, and to express themselves creatively.
- Scratch embeds programming in a social context, making it easy for people to share and collaborate on projects.

The introduction of Scratch to the introductory computer-science course at Harvard led to a sharp reduction in the number of students dropping the course or receiving a failing grade, and a marked increase in the retention of female students (Malan & Leitner, 2007). There have been similar results in pre-college courses. The National Center for Women & Information Technology (NCWIT), in a case study about Scratch, calls Scratch a "promising practice" for increasing gender diversity in IT. The NCWIT study notes that Scratch "uses hands-on, active learning; it is visually appealing; it allows users to express their own creativity and to build on their own experiences; it gives immediate, understandable feedback; and it allows users to avoid syntax errors without focusing on minutiae, freeing them to focus on processes and concepts."

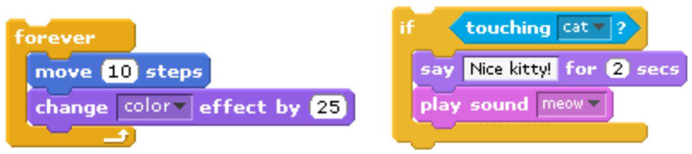
As we develop Scratch 2.0, we will pay special attention to the role it can play in broadening participation in computer science. Just as the multimedia features and expressive possibilities of the current Scratch software have attracted some young people who would not otherwise be interested in computer science, we expect that the social-media and collaboration features of Scratch 2.0 will play a similar role in attracting a broader and more diverse collection of students to programming and computer science. (Of course, Scratch 2.0 will retain the multimedia features and expressive possibilities of the current Scratch version; our goal is not to replace these features but to supplement them.)

3. Results from Prior NSF Support

This proposed project builds directly on the results of our NSF-ITR grant (ITR-0325828) that funded the initial development and study of the Scratch programming environment and website. The project, which ran from 2003-2008 with a total budget of $1,957,000, has received recognition from the educational research, computer science, and design communities. In 2008, Scratch was awarded the Eliot Pearson Award for Excellence in Children's Media, and also an Honorable Mention in the Digital Communities category of Prix Ars Electronica.

Research studies based on the original NSF grant have provided insights into how young people learn important computational concepts while programming with Scratch (Maloney et al., 2008; Resnick et al., in press). We identified three core design principles contributing to the success of Scratch. Compared with other programming environments, Scratch is:

- ***More tinkerable***. To create programs in Scratch, you snap together graphical “programming blocks,” just like LEGO bricks or puzzle pieces. Connectors on the blocks suggest how they should be put together. Just click on a stack of blocks, and it executes immediately. Click on a second stack, and it executes in parallel. People can start by tinkering with the blocks, snapping them together in different sequences and combinations to see what happens.



```
forever
  move 10 steps
  change color effect by 25

if touching cat ?
  say Nice kitty! for 2 secs
  play sound meow
```

Figure 1: Sample Scratch programming scripts

- ***More meaningful***. People are most creative when they are working on personally-meaningful projects. In developing Scratch, we put a high priority on: (a) supporting a wide *diversity* of projects (stories, games, animations, simulations), so that people with widely varying interests can all work on projects that they care deeply about; and (b) making it easy for people to *personalize* their Scratch projects by importing photos and music clips, recording voices, and creating graphics.

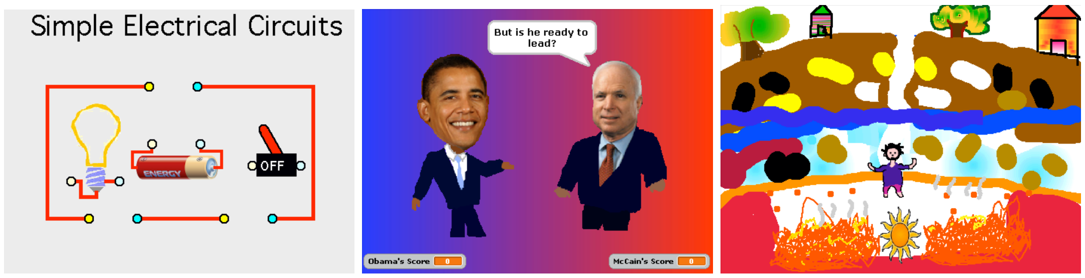


Figure 2: Screenshots from sample Scratch projects, showing diversity of genres

- ***More social***. From the very beginning, Scratch has emphasized sharing and collaboration. We designed the Scratch interface to make it easy for people to upload their projects to the Scratch community website. The website provides both inspiration and audience. By browsing projects on the site, Scratch community members can get new projects ideas and learn new programming techniques. By posting their own projects, Scratchers can get comments and suggestions from others in the community. Many different types of collaboration have emerged on the Scratch website, including:
  - *Remixing*. All Scratch projects are shared under a Creative Commons license, so Scratch community members are free to borrow images and programming scripts from one another’s projects. Roughly 20% of all projects on the website are remixes of other projects. For example, there are several dozen versions of the game Tetris on the website, as community members continually build upon (and improve upon) each other’s versions.
  - *Team projects*. In several cases, members of the Scratch community have formed online design teams, bringing together people with different expertise (e.g., artwork, programming, game design) to collaborate on projects. One of these teams attracted 18 participants, working together for months on a collection of elaborate projects (such as a Halloween-themed adventure game completed in time for the holiday).

- *Consulting*. Some youth in the Scratch community have offered their creative services to other members of the Scratch community. For example, a 12-year-old community member posted a project with a collection of animated characters. In the Project Notes, she explained that she enjoyed making individual characters more than full projects. She encouraged other community members to use her characters in their stories or games, and she offered to make custom animations upon request. Several members of the community responded to her offer and asked her to create specific animations for their projects.
- *Crowdsourcing*. This form of collaboration is, in some ways, the flip side of consulting. Instead of offering services, some Scratch community members request contributions from others in the community. For example, one 13-year-old Scratcher created a series of animated stories based on anime characters. Each week, she uploaded a new project in the series to the Scratch website, like a regular television show. Periodically, she added new characters to the stories. At one point, she got an idea: why not involve the community in the process? She created and uploaded a new Scratch project that announced a contest, asking other community members to design a sister for one of the characters. She received dozens of replies, and selected the best entry to add to her series. Other youth on the Scratch site have organized other types of creative contests.

The social aspects of Scratch have helped attract a diverse community of participants. For example, a 13-year-old girl co-founded a game production studio on the Scratch website, organizing and working with 20 other youth on the development of online games with elaborate story lines. She described her experiences of collaboration:

> *What is fun about Scratch and about organizing a company to write games together is that I've made a lot of friends and learned lots of new things. I've learned a lot about different kinds of programming by looking at other games with interesting effects, downloading them, and looking at and modifying the scripts and sprites. I really like programming!*

4. Technological Infrastructure for Creative Collaboration

Although the current Scratch application and website have inspired many types of collaboration, many members of the Scratch community (as well as members of our MIT design team) feel limited by some of the constraints of the current software. By developing a new generation of Scratch, with the programming application and website more fully integrated with one another, we believe that we can provide an infrastructure for new forms of collaboration – and gain insights into new strategies for supporting creativity and collaboration in online communities.

Our plans for Scratch 2.0 are aligned with the computer industry trend to migrate more applications and services to the "cloud" – that is, providing computation as a service, with applications and infrastructure hosted on dynamically-scalable remote webservers rather than local machines. We are planning to implement the authoring environment of Scratch 2.0 as a Rich Internet Application that runs on a web browser using technologies like Flex/Flash or HTML5/Javascript, hosting the site with an easily scalable platform like Amazon Web Services or the Google App Engine.

Young people are already interacting with many cloud-based services, such as YouTube and Facebook. But Scratch 2.0 is fundamentally different in that it aims to engage people (particularly children and teens) in programming their own projects and activities in the cloud. With Scratch 2.0, people won't just interact with the cloud, they will create in the cloud. The goal is to democratize the development of cloud-based activities, so that everyone can become an active contributor to the cloud, not just a consumer of cloud-based services.

In the rest of this section, we present examples of new types of activities supported by Scratch 2.0 – and how those activities will encourage creativity and collaboration in the cloud.

4.1 Programming with Social Media

Just as the current version of Scratch has engaged youth in programming by building upon their interest in multimedia activities (manipulating graphics, photos, music, and sound), we believe that Scratch 2.0 will further broaden participation in programming by building upon youth interest in social media (e.g., Club Penguin, Facebook, YouTube, DeviantArt, Gaia Online).

With Scratch 2.0, you will be able to create Scratch projects that react to events on social-media websites. It's like adding a collection of "web sensors" to the Scratch programming language, so that Scratch programs can detect and respond to changes on social-media sites. For example, you could create a Scratch project that reacts whenever one of your friends posts an update on their Facebook page or uploads a video on YouTube. Alternatively, you could create a Scratch project that sends out a message to other social-media sites (e.g., sending a Twitter tweet) whenever certain events happen in the Scratch project. For example, you could create a Scratch-based puzzle game that sends out a tweet when someone solves the puzzle.

Many members of the Scratch community are especially interested in events on the Scratch website itself. With Scratch 2.0, they will be able to trigger actions in Scratch projects in response to events on the website. For example, they could create a Scratch project that responds each time someone posts a comment on one of their Scratch projects, or each time someone clicks the "Love It!" button on one of their projects.

Sonya appreciates when other members of the Scratch community write comments and suggestions on her Scratch projects. So she decides to create a special Scratch project to keep track of – and to thank – the members of the community who have contributed the most comments to her projects. The project displays the avatars of the Scratch community members who have contributed the most comments, and continually adjusts the sizes of the avatars based on the number of comments they have contributed. Adapting a sample project she found online, Sonya creates a script with a web sensor that triggers whenever anyone adds a comment to one of her Scratch projects and increases the size of the appropriate avatar in her visualization. She adds another script that slightly decreases the size of each avatar once a week, so that avatars will shrink over time if the associated Scratchers stop leaving comments on her projects.

4.2 Sharing and Visualizing Online Data

The web is awash with data and statistics, but current Scratch projects have no way to access them. That will change with Scratch 2.0. Imagine if your Scratch project could access demographic data from the US Census Bureau website. Or weather data organized by zip code. Or world health statistics from a United Nations database. Or information about your favorite movie stars from the International Movie Data Base (IMDB). Or statistics from every game played in the National Football League. What would you program with that data?

Scratch 2.0 will be designed to handle data in several standard formats, including the popular CSV format, so that people can easily import data from other websites into their Scratch projects.

Janet and Hilary enjoy listening to music together after school, and they are particularly interested in female singers. They often search the web trying to find "unknown" women musicians who are just starting to attract attention. They decide to use Scratch 2.0 to create their own service that tracks female musicians who are on the rise. They learn how to access data from Billboard.com, and they create a script to identify who has jumped the most number of spots on the sales charts from week-to-week. They create another script that searches YouTube for a video of each fast-rising performer and inserts the video into the Scratch project alongside the performer's name. If there are multiple videos of the performer, the script selects the one with the most views on YouTube.

While on a high-school field trip to a local wildlife preserve, Jason and Pedro learn that fish in nearby streams are dying because of pollution from a local industry. Back at school, they create a public service announcement about this problem, and they post the project on the Scratch website. But they want to do more. So they decide to create a Scratch project to help other teens identify similar problems in their own regions. They create a script that asks for a zip code, then searches www.data.gov to find endangered fish species in nearby areas. The script then displays the information integrated into a Google Map of the streams or lakes where the fish live.

4.3 Collaborating through Persistent Data

Of all the suggestions for new features we receive from the Scratch community, probably the most common request is for the ability to save and display the highest scores for Scratch games. That's not possible now, since Scratch projects, even when running on the website, have no history associated with them. Each execution of a Scratch project is independent of all others.

With Scratch 2.0, we plan to provide persistent data structures on the website so that Scratch community members can save data associated with projects, and also share data with one another in shared variables, or "shariables." This feature has implications far beyond high-score lists for games. For example, Scratch community members could create surveys in which they gather information from other members of the online community, store it in the new persistent data structures, and present visualizations of the results (with the visualizations automatically updating as members of the community input more information). We have already seen in this type of project within the Scratch community. For a school science project, a 10-year-old Scratcher created a project that tested reaction times, asking users to click a button when the screen changed color, and then measuring the time between color-change and button-click. The goal was to collect data from Scratch community members around the world and look for correlations between reaction time and other parameters (such as age, weight, nationality, and what sports they play). With the current version of Scratch, there was no easy way to collect this data. With Scratch 2.0, such surveys and experiments will be easy to set up.

Using these new database features, Scratch community members could also create their own to-do lists, calendars, address books, or even simple blogging systems on the web. And by sharing data structures with others, they could create simple chat systems.

Christina, age 14, enjoys using TheJungle, a commercial website where she can represent herself as an avatar (selected from list of jungle animals) and chat with her friends. She likes TheJungle, but wishes she could create her own customized version for herself and her friends. She logs into Scratch 2.0, and clicks a button to create a list which will hold chat messages. She creates a simple script that asks people to type in a chat message, then puts their responses into the list. She contacts her friend Anna and asks her to test it out. Once Anna and Christina confirm that it works, they decide to create a more complicated system that lets people create their own custom-designed interactive avatars, which will be displayed alongside their chat messages in the project. As other friends join in, they begin inventing a story-based role-playing game with fantasy animals, played through their new chat program.

4.4 Sharing at Multiple Granularities

On the current Scratch website, members of the community can share Scratch projects with one another, but the website does not provide any easy way to share parts of Scratch projects, such as sprites (the characters in Scratch projects), programming scripts, images, or sounds. Members of the Scratch community have tried to work around this limitation in several ways. Some community members have posted projects with nothing but a collection of animated sprites, explicitly encouraging others in the community to use and modify the sprites. Joren Lauwers, a 14-year-old member of the Scratch community (and a member of the Advisory Board for the Scratch 2.0 project), designed a separate website dedicated solely to the sharing of Scratch sprites and media.

With Scratch 2.0, we will support sharing at multiple levels of granularity, enabling members of the community to share all different types of Scratch-related objects (sprites, scripts, images, sounds), and to create and share libraries with collections of objects. We will also develop better ways for community members to search through objects on the Scratch website. Within the 500,000 projects currently shared on the Scratch website, there are more than 4 million sprites and 13 million programming scripts and an even larger number of graphic images, but there is no way to search through all of these assets. Scratch 2.0 will provide ways to help you find the assets you need for particular projects. For example, if you are in the process of creating a script or sprite, you will be able to ask the system to show you all other scripts or sprites (from the millions on the website) that are similar to the one that you are creating.

5. Design Experiments for Creative Collaboration

As we develop Scratch 2.0, we will continually run *design experiments* to examine how our design choices influence activities and attitudes in the Scratch online community. In these experiments, we will release new features to the Scratch community, then study (both qualitatively and quantitatively) how the community responds to the changes. In some cases, we will release alternative versions of the features to different subgroups within the community, then study the differences in how the subgroups respond to the alternative versions.

This type of experimentation is greatly facilitated by the transition to a web-based version Scratch. Previously, we released new versions of the Scratch programming application only twice a year, since each upgrade required members of the community to go through a lengthy process of downloading and installing the application. By contrast, we have upgraded the Scratch website (as opposed to the programming application) much more frequently. With Scratch 2.0, all software will be embedded in the web, so we will be able to release new versions of the Scratch programming application much more quickly and easily, and the entire community will have access to the new versions instantaneously. Thus, it is possible to go through much faster iteration cycles of designing new features, testing them with the community, then revising the features based on the response from the community.

Before we release Scratch 2.0 to the entire Scratch community, we will set up an alternative website where beta testers can try out prototype versions. For testing certain features, we will work with smaller subgroups of beta testers. For example, we plan to set up an alternative website for computer-science educators, such as David Malan (a member of our Advisory Group) who integrated Scratch into the introductory computer-science course at Harvard, and has expressed interest in experimenting with Scratch 2.0 in his classes.

In the rest of this section, we start by describing some design experiments that we already conducted with the current version of Scratch. Then, we give a few examples of new design experiments we plan to conduct as we develop Scratch 2.0. These examples are not intended to be comprehensive, but rather to give a sense of the types of design experiments we will conduct. Finally, we discuss visualizations that we plan to create (based on the design experiments) to provide a clearer picture of collaborations within the Scratch 2.0 community.

5.1 Previous Design Experiments with Scratch

The Scratch website has already gone through many design iterations, with a steady stream of changes designed to encourage collaboration. Shortly after the website was launched in 2007, we noticed a flurry of comments on Scratch projects accusing the project creator of "stealing" ideas from other projects. We publicly replied to the comments, explaining that we encourage members of the Scratch community to remix and build upon one another's work, as long as they give credit to the original authors of the work. But charges of "plagiarism" and "stealing" continued within the community, and few remixers explicitly gave credit to original authors, so we began to experiment with design changes to the site. We made two significant changes in an effort to change the way community members view remixing:

- We created an automatic attribution mechanism, so that remixed projects automatically include a link to the original project (and original projects include a link to all remixes derived from the project)
- We modified the front page of the site to feature the "Top Remixed Projects," so that having one's project remixed would be seen as a source of pride (and a boost to one's reputation), not as an ethical violation.

These changes led to a statistically-significant increase in the number of remixes shared on the site (increasing from 18% of all projects to 21% of all projects), and a decline in the accusations of stealing and plagiarism.

5.2 Design Experiments with Scratch 2.0

As we move forward with Scratch 2.0, we will continue to study interactions on the website, and we will run experiments to test the effectiveness of various design changes in supporting and encouraging collaboration. We will focus especially on the design lever of "transparency," providing members of the Scratch community with more information about what others are doing, and the histories and reputations of different members of the Scratch community. Our expectation is that people will engage in more and deeper collaborations if they have a better understanding of the people with whom they are collaborating.

Here are examples of the types of design experiments that we plan with Scratch 2.0:

- *How do community norms influence social dynamics?* Currently, members of the Scratch community are not notified when their projects are remixed. We plan to start sending remix notifications, and we will run an experiment to see if the content and tone of notification messages influence the social dynamics around remixing and collaborating. When a Scratch member's project is remixed, the member will be randomly assigned to receive one of two messages. In some cases, they will receive a notification saying "Congratulations! <user-name> has remixed and added new features to your project!" In other cases, they will receive a more neutral message: "Your project has been remixed by <user-name>." We will investigate whether the difference in notification messages influences people's attitudes and behaviors related to remixing and collaborating – for example, do people who receive the more positive notification write more positive comments on remixed projects?
- *How does automated attribution influence collaboration?* With Scratch 2.0, it will be possible to share not only full Scratch projects but many different types of assets within projects (sprites, images, sounds, scripts, even fragments of scripts). We will experiment with keeping an attribution chain for each asset, keeping track of each person who shared or modified the asset. We will investigate whether the public availability of this attribution chain influences the ways people share assets and their attitudes towards sharing.
- *How can we encourage people to give credit when they create remixes?* As described earlier, the Scratch website now automatically generates a link to the original project when someone posts a remix of that project. But we have done some preliminary studies that indicate creators of Scratch projects feel more satisfied (and less hostile to the process of remixing) when they receive acknowledgement directly from people who remix their projects, rather than simply receiving an automated acknowledgement generated by the Scratch website. So we will experiment with design changes in an effort to encourage remixers to give credit to original authors. Currently, the most common way to acknowledge original creators is through the Project Notes associated with each project. We will add separate sections specifically for credit and acknowledgement, to see whether that changes attitudes and behaviors towards remixing.

5.3 Visualizations of Collaborative Activity

In connection with the design experiments, we will develop visualizations to help understand the nature of collaborative activity in the Scratch community, and how collaborative activity changes over time. These visualizations are intended primarily for research purposes, but we expect that they will also be of interest to members of the Scratch community.

For example, we created the following visualization to show the types of comments that Scratch community members leave on projects that are remixes of their own projects. The visualization is based on all two-word combinations used in the 13,000 comments that fit in this category. As you can see, the most common comments are positive (*good job*, *nice remix*), but some are more negative or accusatory (*give credit*). Observing visualization like this over time can provide an indication of whether design changes influence attitudes towards remixing and collaboration.

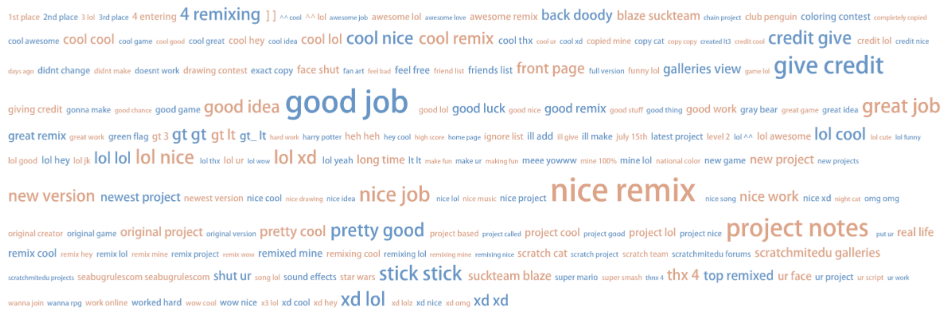


Figure 3: Visualization of comments on remixed projects (comments by original creators)

Here are a few examples of other visualizations that we plan to develop in connection with design experiments:

- Remix trees, showing the "genealogy" of a project, with its remix antecedents and descendents
- Graphs showing how the popularity of projects correlates with the number of collaborators working on the project
- Graphs showing how the complexity of projects correlates with the number of collaborators who worked on the project

6. Dissemination

Our research team has a strong track record for getting our ideas, activities, technologies, and strategies disseminated nationally and internationally. Millions of young people are currently using technologies based on our research (including Scratch and LEGO MindStorms robotics kits), in both formal and informal educational settings. Our team has also developed educational programs with a broad impact: the Computer Clubhouse network of after-school learning centers, founded by two members of our team, has expanded to 100 sites in 20 countries, reaching 20,000 young people in low-income communities (Rusk, Resnick, & Cooke, 2009).

We will make Scratch 2.0 software available for free download from the Scratch website (http://scratch.mit.edu). The website currently attracts more than 500,000 unique visitors each month. More than 1 million people have downloaded the current version of Scratch software, and we expect an even larger number will use Scratch 2.0.

We will disseminate our research results through multiple channels:

- *Papers and Publications*. Members of the project team have a strong record of publishing in academic journals and broader-circulation publications to parents, educators, and the general public. In addition, our group's work is frequently featured on educator websites and mass-media outlets, reaching a broad and diverse audience.

- *ScratchEd*. Earlier this year, our research group launched a new website and online community, called Scratch-Ed (http://scratched.media.mit.edu), specifically for educators using Scratch. We will share research results and educational strategies related to Scratch 2.0 on the ScratchEd site.
- *Hands-on Workshops*. Our team regularly runs professional-development workshops for educators. For example, we organize workshops each year at the Building Learning Communities conference and the National Educational Computing Conference. In 2009, we organized a Google-sponsored Computer Science for High School (CS4HS) workshop, and Google has invited us to continue offering CS4HS workshops in the future.
- *Conference Presentations*. Members of our team have made presentations about Scratch-related research at many major research and educational conferences, including the Creativity and Cognition Conference, SIG-CSE conference, Computer Supported Collaborative Learning conference, American Educational Research Association conference, CHI conference, National Educational Computing Conference, and many others. We will share Scratch 2.0 results at these conferences in the future.
- *Scratch Community Events*. We will highlight Scratch 2.0 (and research related to Scratch 2.0) at Scratch community events such as Scratch Day and the Scratch@MIT conference.

7. Timeline

Year One: Preliminary Design Experiments and Scratch 2.0 Prototype
• Conduct design experiments with current Scratch to inform design of features for Scratch 2.0 • Design Scratch 2.0 prototype based on input from users, advisors, and design-experiment results • Implement prototype version of Scratch 2.0
Year Two: Iterative Refinement and Design Experiments
• Initial testing of Scratch 2.0 infrastructure with beta-testers (both online and in-person workshops) • Conduct series of design experiments with Scratch 2.0 • Iterative refinement of Scratch 2.0 based on findings from design experiments • Develop initial visualizations of collaboration patterns
Year Three: Dissemination of Scratch 2.0 and Research Findings
• Public release of Scratch 2.0 • Ongoing studies of collaboration activity in Scratch 2.0 community • Publish research results and present at national conferences • Educational outreach through series of workshops at conferences and other events

8. Key Personnel

The project will be led by the same core team that designed and developed the highly successful Scratch software and online community (with previous funding from NSF). The team brings together decades of experience in designing innovative educational technologies and creative learning environments for children and teens.

Mitchel Resnick, Professor of Learning Research at the MIT Media Lab, specializes in the development and study of new technologies that engage children in creative learning experiences. His research group developed (with NSF support) the "programmable bricks" that were the basis for the LEGO MindStorms robotics kits, and the Scratch software and website that are the basis for this proposed project. He co-founded the Computer Clubhouse network of after-school learning centers and the NSF-funded PIE Network of museums. Resnick earned a BS in physics from Princeton in 1978, and an MS and PhD in computer science from MIT in 1992. He was awarded an NSF Young Investigator Award in 1993. In the proposed project, Resnick will serve as Principal Investigator and Project Director.

John Maloney, Research Specialist at the MIT Media Lab, has been the lead developer for Scratch project since the start of the project in early 2003, implementing both the main Scratch application and the Java player that allows Scratch projects to be run in a web browser. Prior to joining the MIT Media Lab, Maloney worked for Alan Kay at Apple Computer and Walt Disney Imagineering. Maloney earned BS and MS degrees in computer science from MIT in 1981 and a PhD in computer science from the University of Washington in 1991. In the proposed project, Maloney will serve as co-PI, leading the development of Scratch 2.0 software.

Natalie Rusk, Research Specialist at the MIT Media Lab, focuses on the development of technology-based programs that build on young people's interests. She is a core member of the Scratch design team, and has led the development of educational materials for the Scratch project. She co-founded the Computer Clubhouse after-school program and led the development of the Learning Technologies Center at the Science Museum of Minnesota. She served as Network Director of the NSF-funded PIE Network, a collaboration with six museums to develop a new generation of creative hands-on science activities. She has a Master's degree in Interactive Technology from Harvard Graduate School of Education, and is pursuing her PhD in Child Development at Tufts University. In the proposed project, Rusk will serve as co-PI, leading the educational aspects of the project, with a focus on ensuring that Scratch 2.0 addresses the needs and interests of a broad diversity of young people.

Andrés Monroy-Hernandez, PhD student and Research Assistant at the MIT Media Lab, leads the development of the Scratch website and online community. His research focuses on the design and analysis of social platforms to support creative and collaborative learning. He holds a bachelors degree in Computer Science from Tec de Monterrey in México and a masters degree from the MIT Media Lab. Before coming to MIT, he worked for four years as a software engineer on library-automation projects. In the proposed project, Monroy-Hernandez will focus especially on developing social-media applications of Scratch 2.0 and studying how design decisions on Scratch 2.0 influence cooperation within the online community.

9. Advisory Board

We have assembled an exceptional and multidisciplinary Advisory Board, including computer scientists, educators, social scientists, and designers. The Advisory Board also includes two teens who have been active participants in the Scratch online community.

Yochai Benkler is Berkman Professor of Entrepreneurial Legal Studies at Harvard University and faculty co-director of the Berkman Center for Internet and Society. Benkler is a leading theorist of collaboration and cooperation on digital networks, with a focus on how peer production and sharing can have transformative effects on the economy and society.

Kate Bielaczyk is an Associate Professor in the Learning Sciences Lab at the National Institute of Education in Singapore. She is an expert on collaborative knowledge building in school classrooms, and has been an important contributor to research on Knowledge Forum, a shared-discourse software environment that supports knowledge-building communities.

Amy Bruckman is an Associate Professor at the College of Computing at Georgia Tech and member of the NSF-funded Broadening Participation in Computing alliance in Georgia. Bruckman specializes in research on online communities and education, and is currently studying tools for scaffolding creative online collaboration.

Mark Guzdial is a Professor in the College of Computing at Georgia Institute of Technology. Guzdial is a leading researcher in the field of computer science education and developer of the Media Computation approach to introductory computing. He is currently serving as Vice-Chair of the ACM Education Board, contributing to reform in high school computer science education.

Joren Lauwers (Scratch username: JSO) is a youth member of the Scratch community and serves as one of the moderators of the Scratch online forums. On his own initiative Lauwers created a website for sharing Scratch sprites and scripts, which was included as a featured resource in the mass-market book *Scratch Programming for Teens*.

David Malan is a Lecturer on Computer Science at Harvard College for the School of Engineering and Applied Sciences. Malan integrated use of Scratch programming into the introductory computer science course at Harvard, and has published research on how Scratch has contributed to changes in student retention rates and attitudes.

Geetha Narayanan is Founder and Director of the Srishti School of Art, Design, and Technology in Bangalore, India, and is an international leader in innovative design education. Narayanan is using Scratch as a tool for creative expression and empowerment among youth in Drishya community centers located in impoverished slum neighborhoods in Bangalore.

Lila Pustovoyt (Scratch username: MyRedNeptune) is a youth member of the Scratch community and one of the moderators of the Scratch online forums. Pustovoyt has initiated a variety of creative collaborations on the Scratch website, and was an important contributor to a Scratch “company” that brought together youth from five different countries.

Keith Sawyer is an Associate Professor of Education at Washington University, and an international leader in the study of creativity and innovation. He is the author of *Explaining Creativity: The Science of Human Innovation* and the editor of *The Cambridge Handbook of the Learning Sciences*.