Will it Stick? Exploring the Sustainability of Computational Thinking Education Through Game Design

Kyu Han Koh
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 303 492 1349
kohkh@colorado.edu

Alexander Repenning
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 303 492 1349
ralex@cs.colorado.edu

Hilarie Nickerson
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 303 492 1349
hnickerson@colorado.edu

Yasko Endo
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 303 492 1349
yasko.endo@colorado.edu

Pate Motter
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 303 492 1349
Pate.Motter@colorado.edu

ABSTRACT
Exposing middle school students to computer science through game design appears to be a promising means to mitigate the computer science pipeline challenge. Particularly when game design is integrated into already-existing middle school courses, either with an academic or a technology skills focus, there is a high potential for exposure, often reaching hundreds of students per school per year. In contrast to after-school programs, students taking these classes are usually not self selected, creating an important opportunity to vastly broaden participation in computing activities and to include more female and underrepresented students. Research suggests that exposure to short game design activities is effective in motivating large percentages of students in a wide variety of demographic groups. The Scalable Game Design project has trained middle school teachers around the US to teach students how to make one simple arcade-style game in a 1-2 week session. A study with over 10,000 students is exploring the sustainability of this approach and finding positive responses to inquiries such as these: Do teachers continue to use game design? Do they have the desire and capacity to move forward without extrinsic rewards such as research stipends? After building one game, do students advance to building more games or even STEM simulations?

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computers and Information Science Education

Keywords
Sustainability, Game Design, Simulation Design, Computational Thinking, Zone of Proximal Flow.

SIGCSE'13, March 6–9, 2013, Denver, Colorado, USA.

1. INTRODUCTION
The need to start early with computational thinking (CT) activities in middle school has been widely recognized [4]. Recently, the number of students selecting computer science as a field at the college level has gradually been going up, but the same cannot be said of the participation of women and underrepresented students. Activities such as game design [11, 12], robotics [8], story telling [15] and animation [11] have mostly focused on the motivational angles that are important to address given the rather negative perception of computer science. Many students find that "programming is hard and boring." For some time, the Scalable Game Design (SGD) project has been a proponent of the idea that it is not only important but also quite feasible to integrate CT into public school education [1]. SGD introduces middle school students to CT through game design. Particularly when game design is integrated into already-existing middle school courses, either with an academic or a technology skills focus (e.g., keyboarding, use of office applications), there is a high potential for exposure, often reaching hundreds of students per school per year. Moreover, in contrast to after-school programs, students taking these classes are usually not self-selected. Consequently, this strategy affords an important opportunity to vastly broaden participation by including more female and underrepresented students. For example, research suggests that exposure to short game design activities is effective in motivating large numbers as well as large percentages of women and underrepresented students with respect to interest in computer science [16]. In past surveys, the percentages of students who desired to participate in more game design were as follows: 64% female / 74% male; 69% underrepresented / 71% white [16].

To assess the efficacy of a CT education strategy, it is important to look beyond just tools, activities and curricula. High levels of motivation [16] are most likely essential in achieving efficacy but certainly not sufficient alone. Aside from motivational angles, can game design activities contribute to the core education challenges connected to STEM education? Can teachers with little, if any, programming background be trained to teach students how to design and implement games? Middle schools in particular usually have very little time and even fewer resources for

professional development, especially given that CT/computer science/programming is simply "not on the test" and consequently does not factor into strategically important school assessments such as the No Child Left Behind act.

The sustainability of an instructional strategy is a more robust indicator for efficacy but at the same time also harder to evaluate than motivational and educational goals. Implementers must overcome potentially strong novelty effects [2] and address a number of other concerns that typically require longitudinal research to discover. It is often easy to find a number of highly motivated teachers and students to initially participate in some kind of experiment, but what will happen if support structures including the presence of assistants in the classroom or perhaps even financial support and training fade and the research ends? Will the teachers want to continue? Do they have the means to continue? Can students move beyond basic activities? Does school management appreciate these efforts?

This paper compares the theory and practice of the Scalable Game Design project with respect to sustainability. To better interpret the outcomes, it will help to provide a brief introduction to the Scalable Game Design *theory of change*. Designed with sustainability in mind, its essence is that programming challenges and skills should be balanced, providing different paths along which students can increase their skills in order to tackle more advanced challenges. The four core principles of this theory of change [13] address not only motivation, but also scaffolding for schools that is intended to stimulate ongoing interest by administrators, teachers, and students in continuing these CT skill-building activities. These four principles are captures in <approach> *so that* <outcome> format:

1. ***Exposure:*** Develop a highly adoptable middle school CT curriculum integrated into existing computer education and STEM courses *so that* potentially large and diverse groups of children are exposed to CT concepts.
2. ***Motivation:*** Create a scalable set of game design activities ranging from low-threshold to high-ceiling activities *so that* students with no programming background can produce complete and exciting games in a short amount of time, enticing them to move on a gradual trajectory to the creation of highly sophisticated games and STEM simulations.
3. ***Education:*** Build computational instruments that analyze student-produced projects for CT skills *so that* learning outcomes can be objectively measured.
4. ***Pedagogy:*** Systematically investigate the interaction of pedagogical approaches and motivational levels *so that* teachers can broaden participation more strategically.

The resulting Scalable Game Design framework (Figure 1) [3, 6] includes a curriculum of increasingly advanced game design activities that range from basic classic 1980s arcade games such as Frogger (bottom-left in the figure) to more contemporary games such as The Sims. As students progress, they encounter sophisticated concepts such as artificial intelligence (top left in the figure). The right-hand side shows the Zones of Proximal Flow diagram—a combination of Csíkszentmihályi's Flow diagram with Vygotsky's Zone of Proximal Development conceptualization [16]. Project-First learning, a just-in-time approach of introducing skills in context as needed, sustains high student motivation levels throughout the learning process.

To measure the efficacy of this approach, student learning is evaluated by employing a tool called Computational Thinking Pattern Analysis (CTPA) [5]. Every game and simulation produced by students—more than 10,000 over the last 4 years—is not only collected in the Scalable Game Design Arcade [6], but is also analyzed with respect to CT thinking skills expressed by students. CTPA is not looking for constructs such as IF and LOOP statements at the programming level but, instead, is looking for more general object interactions such as collisions and diffusion at a phenomenalistic [9] level through the use of Latent Semantic Analysis [7] inspired methods to find code patterns. These phenomenalistic patterns, Computational Thinking Patterns [3], can be understood as a subset of universal CT skills that are relevant to game design as well as to other applications including the creation of science simulations.

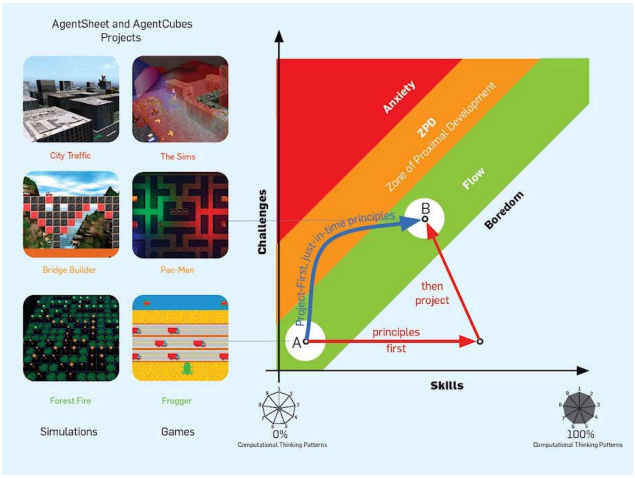


Figure 1. Zones of Proximal Flow: The Scalable Game Design framework includes a CT curriculum that balances CT challenges with CT skills. A Project-First learning path maintains motivation while students advance from basic skills to more advanced ones.

The combination of exposure, motivation, education, and pedagogy afforded by the Zones of Proximal Flow framework provides a favorable environment for sustainability, as it offers an incremental approach for CT skills development that could transform school systems gradually. In other words, it should be possible for schools to get started quickly and then to move along somewhat predicable trajectories toward more advanced game design or STEM simulation building challenges.

So far this has been the theory. The rest of this paper explores the question of sustainability by analyzing data from schools around the USA that are engaging in Scalable Game Design. Specifically, it investigates whether students mentored by teachers with basic training in SCG principles will move beyond the basics, advancing to more sophisticated game design or even applying their skills to non-game design applications such as STEM simulation building. This outcome would be particularly exciting, as it could be interpreted as providing evidence for the transferability of CT skills that are relevant to game design as well as to simulation-building applications.

2. METHOD

The project discussed in this paper was designed to increase students' interests in computer science and STEM by integrating learning about computational thinking with game design at the

middle school level. To serve a wide spectrum of communities, this project has worked with more than 80 teachers and 10,000 students from urban, rural, suburban and remote/tribal regions in 9 different states.

Common factors for the project schools are as follows:

- Most students were 6th/7th graders.
- Teachers taught Scalable Game Design to students using the AgentSheets game/simulation programming software, employing the Project-First approach (see Figure 1). They attended a week-long training before their SGD implementation.
- The project offered an official game design curriculum for participating schools, and this curriculum started with a basic design for the game of Frogger.
- At the end of each 1–2 week module, during which students completed either a game or a simulation, the students' creations were uploaded to the Scalable Game Design Arcade (SGDA) [6].
- A fixed stipend was given to teachers who finished one module of any project (game or simulation) per semester. There was no extra incentive for implementing additional modules. This meant that teachers could stay with the basic Frogger implementation to receive their stipend, with no obligation for further levels of implementation.

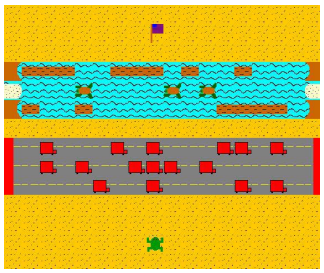
2.1 Class Implementation

The participating middle school classes are exposed to the Scalable Game Design project curriculum in separate class modules. Each module's curriculum includes one game or simulation. Before each class of students begins its first module, a survey is administered to collect demographic and experience level data. A similar post survey is administered after the games have been uploaded. The middle school course trajectory is structured along a difficulty-oriented pipeline. In other words, from a computational thinking perspective, the second game in the curriculum is more difficult than the first, incorporating new CT patterns, and the third learned game increases in difficulty from the second. The course difficulty continuum also deliberately builds on the acquired knowledge and skills from previously learned game tutorials. Within these parameters, the project teachers are free to present the curriculum in any manner that is comfortable for them, as long as the curriculum integrity is adhered to.

2.2 Game Implementation Trajectory

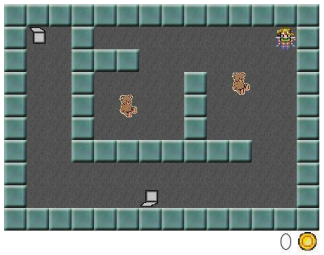
There are several official tutorials for game design on the Scalable Game Design Wiki [14]. Generally, students are guided to learn general programming patterns through implementing a game of Frogger. After Frogger, students are encouraged to learn more sophisticated programming patterns with more complex games such as Pacman, Space Invaders, or Sims.

The project provides several possible learning paths with game implementations. Each learning path is designed to balance CT challenges in a given game implementation and a student's CT skills. For example, students could learn different computational thinking patterns through a "Frogger–Sokoban–Sims" path. Project implementation complexity is designed to increase over time. The paragraphs below show an example of Frogger-Sokoban-Sims learning path followed by one of the project schools, with actual student-created game screenshots.



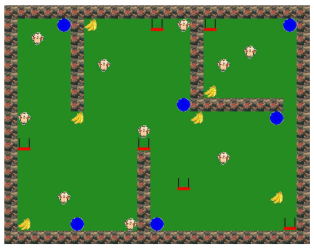
1st game: Frogger

Frogger is typically the first game implemented in the project curriculum. In this simple 80s arcade game, the interactions between agents are relatively simple. Examples include move, hit, and kill.



2nd game: Sokoban

After implementing Frogger, students are asked to implement a game of Sokoban, a classic puzzle game that requires slightly more complicated agent interactions than Frogger.



3rd game: Sims

A game of Sims is considered a good stepping-stone between game design and simulation design. Sims adopts some artificial intelligence techniques such that agents in the game interact with other agents without user control.

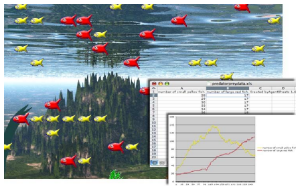
2.3 Simulation Implementation Trajectory

There are several popular STEM simulation models such as the Virus Contagion model, which shows how a virus could be spread; the Forest Fire model, which simulates forest fire; and the Eco System model, which describes food pyramids in ecosystems.

The STEM simulation examples below were created by students from one of the project schools.

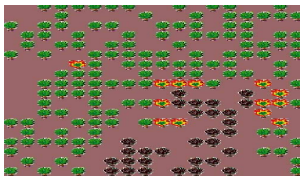
Virus Contagion:

This simulation is often used as an introductory simulation since it requires fewer CT patterns compared to other simulation designs.



***Eco System*:**

There are several different types of Eco System designs, all based on the Predator-Prey model.



***Forest Fire*:**

This simulation includes mathematical techniques such as calculating probability to estimate the direction of fire spreading.

The implementation complexity could be increased by the teacher, for example by adding multiple levels of reproduction rate, animal life spans, and predator-prey relationships.

The project offers basic implementation tutorials for these popular simulation designs. However, the project does not recommend a specific learning path for simulation design. Teachers can design their own STEM simulation or teach any predesigned simulation offered by the project based on their goals and abilities. Designing and teaching original STEM simulations would require extra effort by the teachers.

3. Results: Exploring the Sustainability

Fostering computational thinking through video game camp, student summer camp, computer workshops, and/or after-school programs has been successful [1]. However, there are few reports showing whether those programs are sustainable over a long time period. To validate a program's CT education benefit, the program should show a sustainability of learning over a period of time; otherwise the program's education benefit would end after one experience with the program. The core benefit of computational thinking education lies in bridging problem solving skills between two or more different problem domains [17]. Through our project, we have witnessed many project schools that were able to extend and transfer their students' learning abilities and problem solving skills to the next level of problem domains.

In the following section, we illustrate three kinds of evidence of sustainability: probability to advance, number of different projects, and advancing from game design to simulation design.

3.1 Sustainability: Probability to Advance

Over the last 3 years, 72 different types of games and simulations have been collected from 46 participating schools. All 46 schools submitted at least one project (game or simulation) to the SGDA, and 37 schools submitted two projects or more. Also, 30 schools and 23 schools submitted 3 and 4 projects or more, respectively (Figure 2). Interestingly, these numbers show that 0.8 appears to be a threshold to move forward: 81% of the schools that submitted at least one project, submitted two projects or more, and 80% of this second group submitted three projects or more.

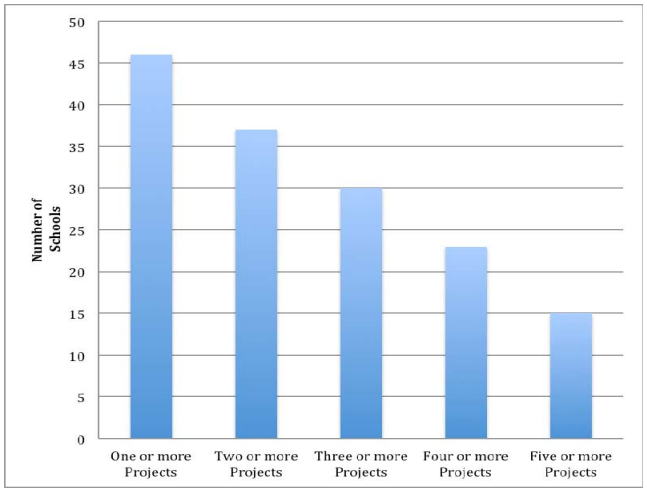


Figure 2. Probability to Advance: A high degree of sustainability is suggested by a large rate of advancement. Over 80% of schools advance to create a second project, of these 80% advance to a third project, of these 80% advance to a fourth project.

Considering teachers' short training timeframe and the lack of financial support after the first module implementation, the 80% success rate can be considered quite high, implying the existence of sustainability. Also, this result possibly indicates that many project schools have successfully helped students follow the Zone of Proximal Flow, spanning students' problem solving skills over multiple problem domains.

3.2 Sustainability: Number of Different Projects

The 46 schools that actively participated in this project submitted students' projects to the Scalable Game Design Arcade. Those schools can be grouped by school locale types. For this project, 9 Tech Hub schools, 22 Urban-City schools, 12 Rural schools, and 3 Tribal schools participated. For school locale definition, we referenced the National Center for Education Statistics (NCES) homepage [10].

Following NCES, the school locale types are defined as below:

- ***Tech Hub:*** Schools within a city with a university and high density of tech support
- ***Urban-City:*** Schools in or very near a city, suburb.
- ***Town:*** Schools less than or equal to 10 miles from urbanized area.
- ***Rural:*** Schools that are less than or equal to 25 miles from an urbanized area.
- ***Tribal/Remote:*** Schools from Ignacio, CO and Oglala, SD, and Native American Indian reservations of Southern Ute and Oglala Sioux.

When project submission data is categorized by region type, schools in tech hubs submitted 46% more projects per school than those in the other three types of regions (Figure 3). However, schools in these region types show similar outcomes to each other. Unique project submission per school in a tech hub is 5.3 while there are 3.6 unique project submissions per school in an urban-city, 3.3 unique project submissions per school in a rural and 3.6 unique project submissions per school in a tribal/remote area.

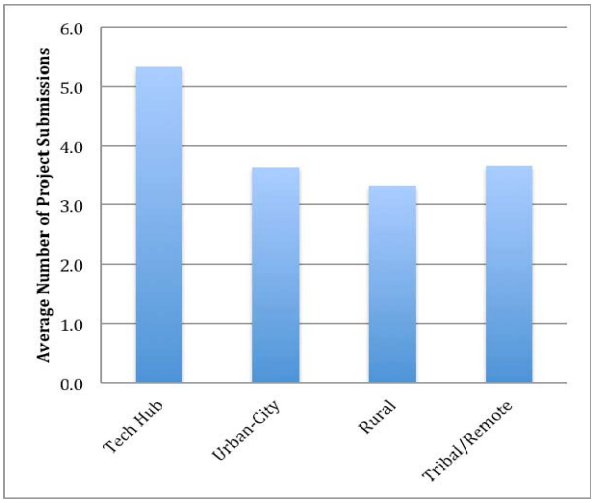


Figure 3. Average Number of Different Project Submissions by School Locale Type

There is one special case that has very large after-school programs and/or summer camp programs in a tech hub area. This

organization, called *Girl Start*, submitted 12 different games and simulations. When this organization is excluded from the data, the number of unique project submissions per school in a tech hub goes down to a value of 4.8. Additionally, there is a second tech hub area school that has been working with the project for over 10 years, with a highly skilled teacher using a specialized curriculum for game design and simulation design. This school produced 17 different types of games and simulations. If those games and simulations were not counted, then the number of project submission per school in a tech hub area would be 3.14, which is similar to other three school areas. However, for clear data interpretation, those two organizations are not excluded in any graph or chart.

3.3 Sustainability: Advancing from Game Design to Simulation Design

There are several good strategies to bring computational thinking into student learning activities [17]. Computational thinking could be taught through an activity of transferring problem solving processes from one problem domain to a wide variety of other problem domains. Thus, when a student transfers acquired problem-solving skills from game implementation to simulation implementation, we could say that evidence of a student's ability to use computational thinking emerges. This seems to be related to the sustainability of computational thinking education since the learning continues toward a higher level of problem domain instead of stopping at the same level of problem domain.

For example, one student created a science simulation based on chaos theory with computational thinking patterns that he learned from implementing Frogger and Sokoban [5].

The project's official game design curriculum includes simulation implementation continued from the basic game implementation. The project strongly encourages teachers to move toward simulation implementation for bridging basic programming education and STEM education, but it is not required for teachers. There is no incentive for simulation implementation.

Regardless of the lack of further financial incentive and a more advanced level of implementation, 43% of project schools successfully moved toward simulation implementation (Figure 4).

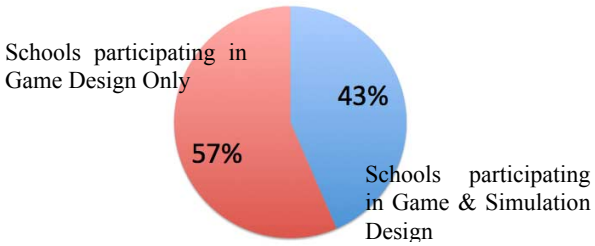


Figure 4. Game Design vs. Game & Simulation Design

In this result, schools in a tech hub area also showed better performance than those in the other three areas. Of the schools in a tech hub area, 66.7% have progressed into simulation implementation. Fewer than half of the project schools not in a tech hub area progressed in this way (Figure 5).

Every project teacher was aware that there was no project incentive for the second module implementation, more advanced levels of implementation, or simulation implementation. Nevertheless, project teachers were willing to move beyond the basic game design as the graphs show.

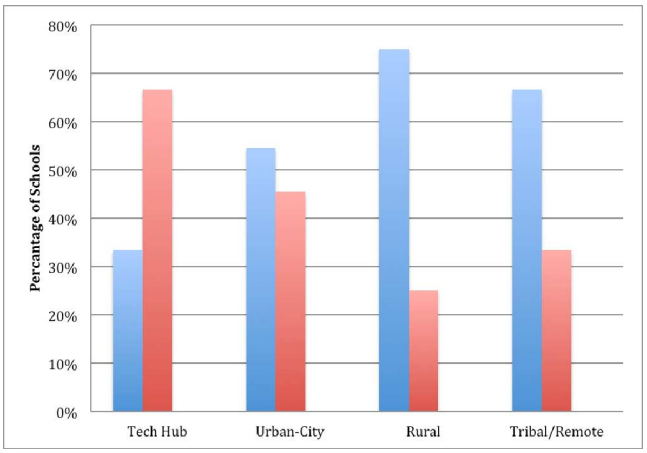


Figure 5. Game Design vs. Game & Simulation Design by School Locale Type

4. DISCUSSION

The large percentage of schools that showed evidence of curricular sustainability by expanding their Scalable Game Design offerings beyond the introductory lessons (80%) is somewhat remarkable given both the typical fate of trial STEM curricula and the many pressures faced by educators. We believe that a combination of student-, teacher- and school-related factors is partially responsible for this outcome. Motivation surveys completed by students in the first two years of the project, which were administered following the completion of SGD activities in their classes, showed that approximately two thirds of the participants would be interested in taking another game or simulation design course and that only 6% offered negative comments regarding their initial experiences [16]. In addition, the number of previously trained teachers who have returned to our Summer Institute for advanced instruction suggests that educators also find value in what the students are learning. In 2012, this professional development opportunity emphasized STEM applications and the design of three-dimensional games. With positive feedback from both students and teachers, schools that have the resources to do so may choose to increase the scope of their Scalable Game Design activities by adding more advanced games and/or promoting the transition to simulation design. It is likely that the higher simulation-to-game ratio of Arcade submissions observed in certain Tech Hub schools versus other types of schools reflects a greater ability to integrate SGD longitudinally into their curricula. Given more time, schools that do not currently have the resources to expand their activities as quickly may show similar patterns of adoption.

Ongoing improvements in how we train and support teachers may also play a role in encouraging schools to proceed beyond basic game design activities. Previous research suggested that guided discovery would be more effective than teacher-directed instruction at increasing interest by girls in pursuing future game design activities, narrowing or eliminating the gender gap even in classrooms where boys significantly outnumber girls [16]. Therefore, for the past two years, our training for teachers new to the SGD curriculum has included demonstrations of alternative teaching styles and a discussion of the efficacy of the guided discovery approach. Classroom observation has shown that

growing comfort with the AgentSheets software also affects teacher behavior, with teachers who desire to employ the guiding strategy becoming less prescriptive over time. Greater student interest resulting from these adjustments, coupled with increasing ease in curriculum delivery, could certainly lead to schoolwide sustainability and the expansion of SGD activities. The availability of more and better tutorials may also be a contributing factor for schools that submit several different kinds of games and/or simulations to the arcade.

5. CONCLUSIONS

The use of game design as an instrument to introduce computational thinking early on in K–12 education is gradually emerging as a promising practice, particularly when trying to broaden the participation of female and underrepresented students. Sustainability trends are an important indicator of whether game design activities in schools will remain one-shot activities kept alive largely through research support or will be able to be adopted in the practice of public schools. The Scalable Game Design strategy, outlined in this paper, has been designed with sustainability in mind. The data collected over a period of four years and with more than 10,000 games and simulations produced by students suggests that more than 80% of schools have advanced beyond the basic requirement. That is, teachers and students have created more, and in most cases more advanced, games and simulations than they were trained to do. This is particularly impressive given that these teachers did not receive any financial support to do so. Moreover, we believe that the large percentage of schools that moved on from game design to science simulations (43%) is a positive indicator that there is more going on than just game programming. Instead, the skills acquired through game design could be considered at least an important subset of computational thinking.

6. ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under Grant Numbers DLR-0833612 and IIP-0848962. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Alexander Repenning is a founder of AgentSheets Inc.

7. REFERENCES

[1] Bennett, V., Koh, K. H., Repenning, A. 2011. CS Education Re-Kindles Creativity in Public Schools. *In Proceedings of ITiCSE '11: Annual Conference on Innovation and Technology in Computer Science Education,* Darmstadt, Germany, June 27-29, 2011

[2] Fraenkel J. R., and N. E. 2005. *How to design and evaluate research in education.* McGraw-Hill, Columbus, OH

[3] Ioannidou, A., Bennett, V., Repenning, A., Koh, K., Basawapatna, A. 2011. Computational Thinking Patterns. *In Proceedings of 2011 Annual Meeting of the American Educational Research Association (AERA) in the symposium "Merging Human Creativity and the Power of Technology: Computational Thinking in the K-12 Classroom".* New Orleans, April 8-12, 2011

[4] James J. Lu and George H.L. Fletcher. 2009. Thinking about computational thinking. *In Proceedings of the 40th ACM technical symposium on Computer science education (SIGCSE '09).* ACM, New York, NY, USA, 260-264

[5] Koh, K. H., Basawapatna, A.,Bennett, V., Repenning, A. 2010. Towards the Automatic Recognition of Computational Thinking. *In Proceedings of IEEE International Symposium on Visual Languages and Human-Centric Computing 2010*, Leganés-Madrid, Spain, September 21-25, 2010

[6] Koh, K. H., Bennett, V., Repenning, A. 2010. Inspiring Collaborative Benefits: An Interaction between a Virtual and a Physical Group Learning Infrastructure, *In Proceedings of Western Canadian Conference on Computing Education (WCCCE 2010)*, Okanagan, B.C., Canada May 7-8, 2010

[7] Landauer, T. K., Foltz, P. W., Laham, D. Introduction to Latent Semantic Analysis. Discourse Processes, 25, 1998, 259-284

[8] Lawhead, P. B., Duncan, M. E., Bland, C. G., Goldweber, M., Schep, M., Barnes, D. J., and Hollingsworth, R. G. 2002. A road map for teaching introductory programming using LEGO© mindstorms robots. SIGCSE Bull. 35, 2 (June 2002), 191-201.

[9] Michotte, A. 1963. *The Perception of Causality* (T. R. Miles, Trans.). London: Methuen & Co. Ltd.

[10] National Center for Education Statistics (NCES), 2012. NCES's urban-centric locale categories. http://nces.ed.gov/surveys/urbaned/page2.asp

[11] Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A, Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. 2009. Scratch: programming for all. Commun. ACM 52, 11 (November 2009), 60-67.

[12] Repenning, A. 2000. AgentSheets®: an Interactive Simulation Environment with End-User Programmable Agents. *In Proceedings of Interaction 2000*, Tokyo, Japan, 2000.

[13] Repenning, A. 2012. Programming goes back to school. Communication of the ACM 55, 5 (May 2012), 38-40.

[14] Scalable Game Design Wiki, 2012. Scalable Game Design Tutorial. http://scalablegamedesign.cs.colorado.edu/wiki/Category:Tutorial

[15] Werner, L., Denner, J., Bliesner, M., and Rex, P. 2009. Can middle-schoolers use Storytelling Alice to make games?: results of a pilot study. In Proceedings of the 4th International Conference on Foundations of Digital Games (FDG '09). ACM, New York, NY, USA, 207-214.

[16] Webb, D. C., Repenning, A., and Koh, K. 2012. Toward an Emergent Theory of Broadening Participation in Computer Science Education, *In Proceedings of ACM Special Interest Group on Computer Science Education Conference, (SIGCSE 2012)*, February 29 - March 3, 2012, Raleigh, North Carolina, USA.

[17] Wing, J. M. 2006. Computational Thinking. Communications of the ACM, 49(3), pp. 33-35, March 2006.