Transforming "Hard and Boring" into "Accessible and Exciting"

Alexander Repenning

University of Colorado
Computer Science
Boulder, Colorado, 80309, USA
ralex@cs.colorado.edu

Abstract. Many kids describe their perception of programming to be "hard and boring." Computational Thinking, including programming, is considered by many to be an important skill for the 21st century workforce. However, in order for kids to get interested in Computational Thinking is it essential to understand why kids perceive programming to be hard, i.e., to understand cognitive challenges, and why kids perceive programming to be boring, i.e., to understand affective challenges. Over the last 20 years our Computer Science education research creating Computational Thinking tools such as AgentSheets and AgentCubes, developing curricula and providing teacher professional development systematically explored what we call the *Cognitive/Affective Challenges* space. We have not only found strategies to gradually move from "hard and boring" towards "accessible and exciting" but in the process also developed research instruments to computationally assess cognitive as well as affective challenges. This paper outlines the *Cognitive/Affective Challenges* space, briefly describes Computational Thinking Pattern analysis as cognitive instrument, and illustrates Retention of Flow as affective instrument to assess motivation.

Keywords: Computer Science education, Computational Thinking.

1 The Cognitive / Affective Challenges Space

In the context of Computer Science Education we have systematically explored the challenges keeping participants from transitioning from "Have to" to "Want to" and have created the Cognitive/Affective Challenges space (Fig. 1) conceptualizing learning activities in a two dimensional space [1]. For instance, computing prime numbers using Emacs to write C++ code would typically be considered hard and boring (red point in Fig. 1). The root of this framework goes back to our research on Computer Science Education but the framework is more general in nature and applies to all kinds of subjects. However, to better illustrate these challenges this article will focus on Computer Science Education in order to provide concrete examples. The statement "programming is hard and boring," made about 20 years ago by a young girl when asked what she was thinking about programming, does not suggest a workable trade off but rather a heartbreaking lose-lose proposition. Disappointingly, a recent report

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

by Google [2] lists “hard” and “boring” as the top two adjectives describing women’s perception of programming. This perception helps explain why women do not choose Computer Science as field of study. These persistent concerns can be interpreted as a two-dimensional research space. The “hard” part is a cognitive dimension exploring how programming can become more (or less) accessible. The “boring” part is an affective dimension exploring how programming can become more (or less) exciting. The big question is how does one transform “hard and boring” into “accessible and exciting?”

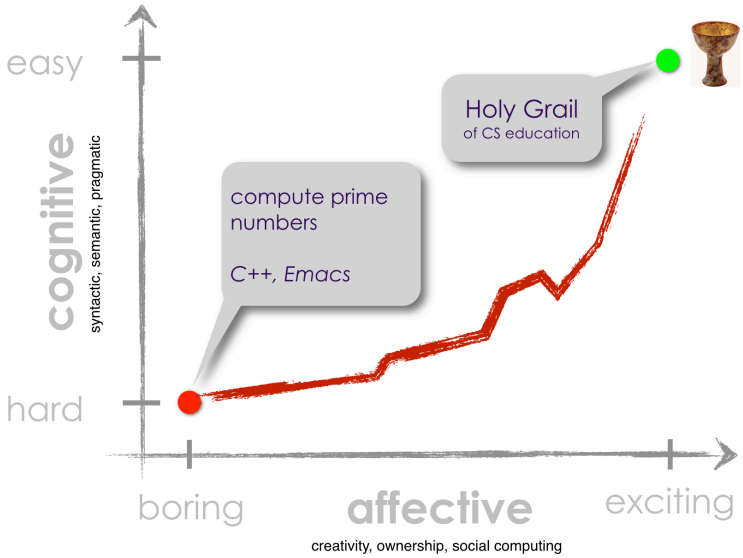

Fig. 1. The Cognitive / Affective Challenges space

The cognitive and affective challenges implied by “hard and boring” can be mitigated with tools, e.g.,

- ***Cognitive Dimension: Making Programming more Accessible through Computational Thinking Tools***. Syntactic, semantic and pragmatic programming issues are examples of cognitive challenges. With visual programming approaches and powerful debugging tools these challenges can be mitigated. For instance, AgentSheets, a programming environment for kids early on [3] pioneered the modern notion of blocks programming through drag and drop interfaces to address syntactic challenges. As AgentSheets evolved over time and new systems such as AgentCubes [4, 5] appeared gradually the notion of a Computational Thinking Tools emerged [6] also addressing semantic and pragmatic aspects of cognitive challenges.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

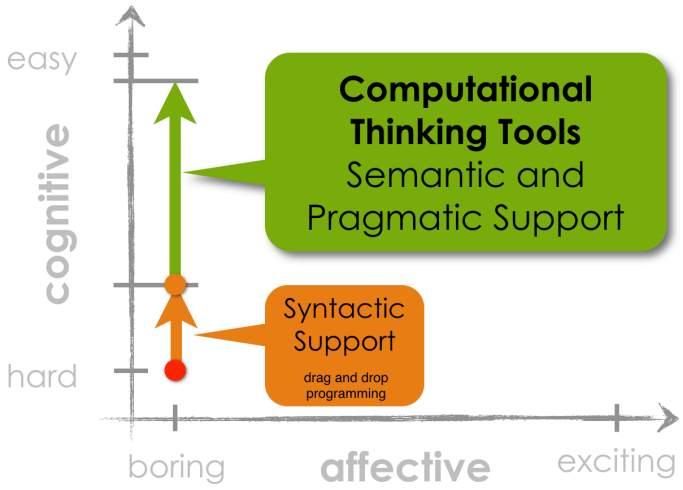

Fig. 2. Computational Thinking Tools address cognitive challenges

- ***Affective Dimension: Making Programming more Exciting through Domain-Oriented Tools***. Domain-Oriented Tools [7] scaffold the creation and programming of interesting content. Using these tools users can create interactive 3D Worlds, robots, and may other things. Key ideas are ownership and creativity (Fig. 3).

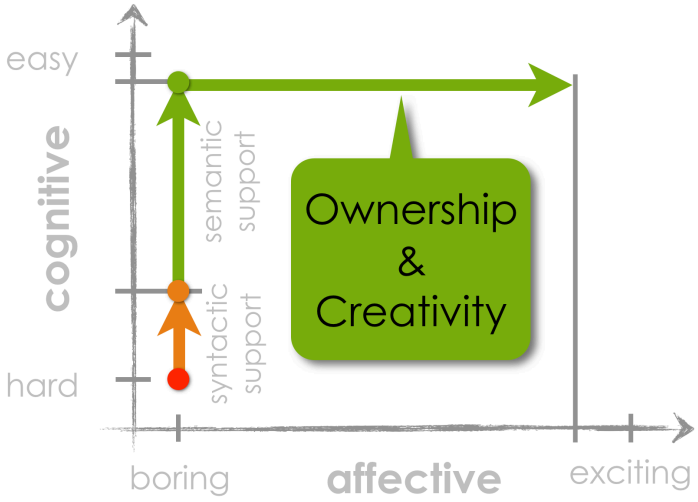

Fig. 3. Ownership and creativity help to overcome affective challenges.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

Some tools such as Inflatable Icons (Fig. 4) focus on the notions of ownership and creativity by providing highly accessible mechanisms to create 2D, 3D and even physical artwork, which then can be turned into programmable games. These ideas can contribute significantly towards the broadening of participation [8].

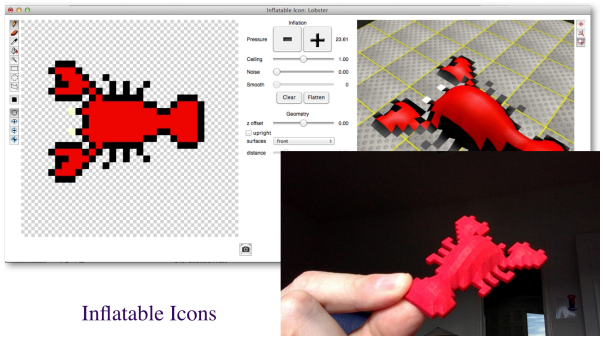

Fig. 4. Inflatable Icons: drawing 2D images, turning them into 3D shapes, and printing them as physical 3D objects

2 Instruments Assessing Cognitive and Affective Challenges

What are indicators that cognitive/affective tools have been sufficiently compelling to transform participation from "have to" to "want to?" The Scalable Game Design project [9] employs game design to get students interested in Computer Science and leverages the competencies acquired to enable students to create STEM simulations. Scalable Game Design employs tools such as AgentCubes online described above to address cognitive and affective challenges. AgentCubes is the combination of a Computational Thinking Tool with a Domain-Oriented Tool. To assess the effectiveness of Scalable Game Design a number of research instruments have been created and tested. One indicator of a transition towards "want to" is the projects created by students. Students do not just create the games they are being instructed to build but they create additional game characters, more levels and even create completely new games, in many cases based on sophisticated programming. In 2013 a 3D Frogger game design tutorial was part of the hour of code and was used by nearly a quarter million participants in just one week. Because AgentCubes online is a cloud based Computational Thinking Tool all the games are available and can be analyzed through educational data mining with respect to cognitive and affective measures:

- ***Cognitive Assessment: Computational Thinking Pattern Analysis (CTPA)***. Computational Thinking Patterns [10-12] describe phenomenalistic object interactions such as the collision of two objects. These patterns can be found in the project code base by computing and comparing high dimensional feature vectors similar to the way Latent Semantic Analysis is used to find matches between snippets of text.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

CTPA can detect patterns independent of application (e.g., game versus simulation). This enables the computation of indicators suggesting transfer of concepts relevant to Computational Thinking from one domain to another. CTPA has been validated and published.

- ***Affective Assessment: Retention of Flow (RoF).*** Retention of Flow uses education data mining to assess Flow of students following instructions to create a game project similar to IKEA customers following instructions to assemble furniture. A Markov-chain model is employed to predict retention functions. With thousands of students building games this model can determine motivational levels, i.e., Flow, over time. Discrepancies between the model and actual data is indicative of potential instructional challenges that may result in either boredom or anxiety. This kind of retention data does exist for 3rd party activities such as the Angry Birds Hour of Code 2013 tutorial created by Code.org. RoF has been used to compare motivational levels across some of these tutorials suggesting very high levels of motivation for Scalable Game Design projects. This research is in an early stage but has already been published [13, 14] including comparisons of Retention of Flow data from different countries [15] (USA, Mexico and Switzerland).

Retention of Flow is particularly relevant to the discussion of "have to" versus "want to" because it is based on voluntary tasks such as programming a game and measures how far participants will progress without being forced. Our 3D Frogger Hour of Code tutorial was intentionally designed to be a cliffhanger activity. That is, within the one-hour time limit of the Hour of Code event participants would be able to create the first couple of agents (the Frog, the road and trucks), create a first game level and program the frog to be controllable by cursor keys. The participants would be able to see that the tutorial has addition chapters raising the question of how likely they would "want to" continue.

In a first stage we just analyzed the retention data exploring how many participants would write at least 1 line of code, at least 2 lines of code, etc. This analysis revealed that the data closely matched negative exponential distributions characterized by survival functions and also found in participation drop off in MOOCs and other programming environments. Discrepancies between the negative exponential fit curve and the actual data (Fig. 2 blue versus black line) could be explained through three types of challenges: cognitive challenges, e.g., confusing instructions, technical challenges, e.g., browsers crashing when typing in unrecognized characters, and practical challenges, e.g., schools only allocating exactly one hour for the activity.

Most interestingly the negative exponential curve continued beyond the first hour of instructions and even beyond all instructions. That is, our cliffhanger approach worked in the sense that participants continued beyond the time typically allocated in schools at rates matching the rates of the first hour. Moreover, the rates were actually higher than the retention rates of the Angry Birds tutorial built by code.org. We can only speculate that the game design process including the design of their own characters and worlds was key to reach these high levels of motivation.

In a second step we wanted to model the decision process that participants must have gone through. Following instructions such as the instructions to build a LEGO

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

construction, e.g., a LEGO Star Wars spaceship, can be conceptualized as sequence of instructions to build an artifact. Each instruction will only be followed with a certain probability. We have formulated a *good design conjecture* suggesting that it makes sense to design tutorials in a way that each step poses roughly the same level of challenge. A Markov chain can be used to model this process of equal probabilities to continue after each step. Using this model we can derive the *probability to continue* from the retention data to find anomalies.

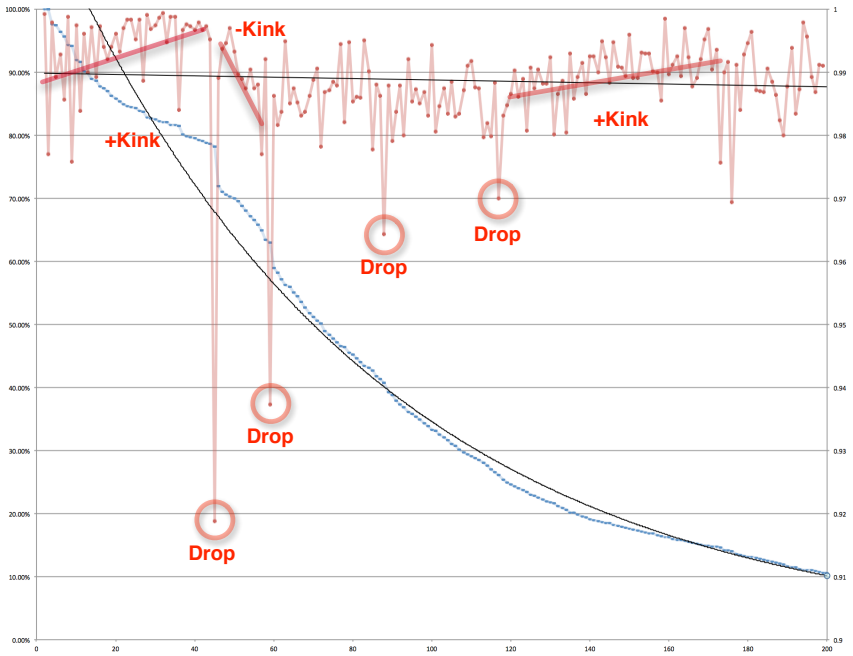

Fig. 5. Retention of Flow (blue) and Probability to Continue (red) versus lines of code. 55 lines of code is roughly the result of one hour of programming.

The data resulting from computing the probability (Fig. 5) to continue is very fine grained and can be used to quickly identify potential trouble spots in the instructional material. These so called drops and kinks can be interpreted as deviations from a Flow state, which, in turn, can shed light on the threshold between "have to" and "want to."

3 Interactions between Cognitive and Affective Challenges

Scientifically speaking it is typically desirable to be able to comprehend individual conditions of complex systems isolated from each other. The Cognitive/Affective space, ideally, would enable the orthogonal investigation of cognitive and affective concerns. However, in reality, it is difficult to separate probable interactions between

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

these two dimensions. For example, it is likely that people are willing to tackle hard problems in the presence of a compelling incentive. These incentives, in turn, may interact with intrinsic and extrinsic motivation [16].

The Cognitive/Affective Space can be segmented into intrinsic motivation (Fig. 6, green = "want to") and extrinsic motivation (Fig. 6, red = "have to"). Extrinsic motivation is often based on rewards such as receiving money for accomplishing a task. However, as suggested by Benabou, the impact of rewards onto intrinsic and extrinsic motivation is surprisingly difficult to predict. For instance, the impact of a reward is likely to depend on the context in which the reward is provided. An "ex ante" award, that is an award that is promised ahead of accomplishing the task may actually diminish intrinsic motivation by signaling that the task will be boring. "Ex post" rewards, in contrast, such as providing a bicycle to a hard working child, may suggest that a task was considered difficult and that the person accomplishing the task exhibited talent. In other words, unlike the "ex ante" reward, the "ex post" reward may have a positive impact onto intrinsic motivation.

Four examples illustrate the "want to" "have to" segmentation illustrated in Fig. 6:

1. ***Writing C++ Program to compute prime numbers***. This is our classical example of a task that is hard and boring. Perhaps with the exception of mathematicians the intrinsic motivation of most people to compute prime numbers is minimal. Moreover, writing a C++ program to compute prime numbers is difficult tasks for people with no prior programming experience.
2. ***Cleaning up your room***. There is no high cognitive load on cleaning up a room but this task is commonly perceived to be very boring. It may be difficult to develop intrinsic motivation for this kind of task due to its Sisyphean nature. It takes considerable effort to clean up the room but the chance that the room will quickly get messy again is high.
3. ***Watch cat videos***. The immense number of people who have watched cat videos on YouTube or even shared cat videos through social media is astronomic and is indicative of how excited most people appear to get by watching cat videos. Watching these videos is certainly not hard. Tasks that are this easy and this exciting, i.e., tasks that are in this deep green section of the "want to" space are not typically well rewarded. In other words, it would probably be difficult to find a job based on tasks like this.
4. ***Solving Crossword Puzzles***. Many find solving crossword puzzles interesting but some crossword puzzles can also be quite hard. The transition from 4a to 4b captures a gradual increase along the cognitive challenge dimension. From Monday to Saturday the crossword puzzle featured daily in the New York Times gradually increases from simple to super hard.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

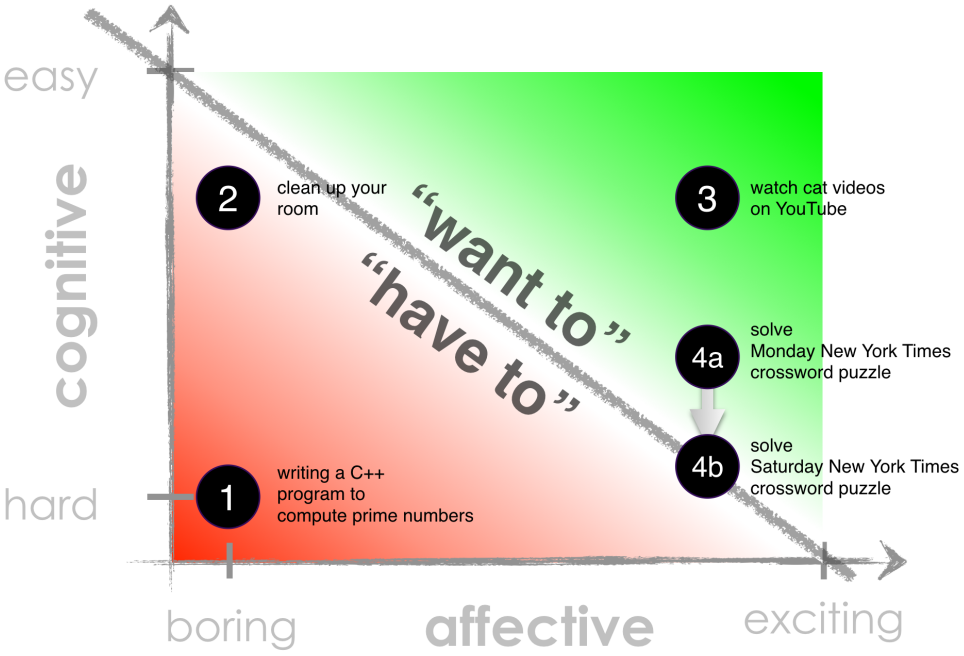

Fig. 6. Tasks: Want to or Have to? The Cognitive and Affective Challenges Interactions

Revisiting Fig. 1 and comparing it to Fig. 6 suggest a potential contradiction. In Fig. 1 the upper right corner, i.e., the Holy Grail of Computer Science education appears to be roughly aligned with "watching cat videos on YouTube" in Fig. 6. Importantly, however, one should not conclude from this alignment that Computer Science education focuses on tasks that are necessarily easy and exciting. Instead, the Holy Grail should be understood as a low threshold starting point for the apprehensive masses with no experience in programming sharing a negative perception of programming. Perhaps the threshold of entry for novices should try to be as low as watching cat videos in order to be become something that people "want to" do. Ultimately, however, Computer Science education must include a well-designed path to move from simpler towards more complex cognitive challenges. Similar to the notion of the gradually more sophisticated New York Times crossword puzzles Scalable Game Design, as Computer Science education strategy, is based on an approach to gradually move from easy to hard Computer Science education challenges [17].

4 Conclusions

The boundary between "have to" and "want to" can be explored through the Cognitive/Affective Challenges framework. On the one hand, tools can be used to mitigate these challenges. On the other hand, instruments can be devised to measure cognitive and affective challenges. There are complex interactions between cognitive and affec-

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

tive challenges but Retention of Flow is a research instrument that can identify and even measure concrete challenges. Once these challenges have been identified through research instruments they can be addressed to improve tools and instructions to gradually shift from activities that are “hard and boring” to ones that are “accessible and exciting.”

5 Acknowledgements

This work is supported by the National Science Foundation under Grant Numbers 0833612, 1345523, and 0848962, by the Hasler Foundation, the Swiss National Science Foundation under grant CRAGP2_158545, and. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of these foundations.

6 References

1. A. Repenning, "Making Programming Accessible and Exciting," *IEEE Computer,* vol. 18, pp. 78-81, 2013.
2. "Women Who Choose Computer Science – What Really Matters, The Critical Role of Encouragement and Exposure," Google, Google Report, Technical reportMay 26, 2014 2014.
3. A. Repenning and J. Ambach, "Tactile Programming: A Unified Manipulation Paradigm Supporting Program Comprehension, Composition and Sharing," presented at the 1996 IEEE Symposium of Visual Languages, Boulder, CO, 1996.
4. A. Repenning, D. C. Webb, C. Brand, F. Gluck, R. Grover, S. Miller*, et al.*, "Beyond Minecraft: Facilitating Computational Thinking through Modeling and Programming in 3D," *IEEE Computer Graphics and Applications,* vol. 34, pp. 68-71, May-June 2014.
5. A. Repenning and A. Ioannidou, "AgentCubes: Raising the Ceiling of End-User Development in Education through Incremental 3D," presented at the IEEE Symposium on Visual Languages and Human-Centric Computing 2006, Brighton, United Kingdom, 2006.
6. A. Repenning, A. Basawapatna, and N. Escherle, "Computational Thinking Tools," presented at the IEEE Symposium on Visual Languages and Human-Centric Computing, Cambridge, UK, 2016.
7. G. Fischer, "Domain-Oriented Design Environments," in *Automated Software Engineering*. vol. 1, ed Boston, MA: Kluwer Academic Publishers, 1994, pp. 177-203.
8. D. Webb, A. Repenning, and K. Koh, "Toward an Emergent Theory of Broadening Participation in Computer Science Education," presented at the ACM Special Interest Group on Computer Science Education Conference, (SIGCSE 2012), Raleigh, North Carolina, USA., 2012.
9. A. Repenning, D. C. Webb, K. H. Koh, H. Nickerson, S. B. Miller, C. Brand*, et al.*, "Scalable Game Design: A Strategy to Bring Systemic Computer Science Education to Schools through Game Design and Simulation Creation," *Transactions on Computing Education (TOCE),* vol. 15, pp. 1-31, 2015.
10. M. Bienkowski, E. Snow, D. Rutstein, and S. Grover, "Assessment Design Patterns for Computational Thinking Practices in Secondary Computer Science: A First Look," SRI International2015.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).

11. A. Ioannidou, V. Bennett, K. H. Koh, A. Basawapatna, and A. Repenning, "Computational Thinking Patterns," presented at the Annual Meeting of the American Educational Research Association (AERA 2011), New Orleans, LA., 2011.
12. K. H. Koh, A. Basawapatna, V. Bennett, and A. Repenning, "Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning," presented at the Conference on Visual Languages and Human Centric Computing (VL/HCC 2010), Madrid, Spain, 2010.
13. A. Repenning, A. Basawapatna, D. Assaf, C. Maiello, and N. Escherle, "Retention of Flow: Evaluating a Computer Science Education Week Activity," presented at the Special Interest Group of Computer Science Education (SIGCSE 2016), Memphis, Tennessee, 2016.
14. A. Repenning and A. Basawapatna, "Drops and Kinks: Modeling the Retention of Flow for Hour of Code Style Tutorials," presented at the The 11th Workshop in Primary and Secondary Computing Education (WiPSCE 2016), Münster, Germany, 2016.
15. N. Escherle, S. Ramirez-Ramirez, A. Basawapatna, D. Assaf, A. Repenning, C. Maiello, *et al.*, "Piloting Computer Science Education Week in Mexico," presented at the Special Interest Group of Computer Science Education (SIGCSE 2016), Memphis, Tennessee, 2016.
16. R. Benabou and J. Tirole, "Intrinsic and extrinsic motivation," *The Review of Economic Studies,* vol. 70, pp. 489-520, 2003.
17. A. Basawapatna, A. Repenning, K. H. Koh, and H. Nickerson, "The Zones of Proximal Flow: Guiding Students Through A Space Of Computational Thinking Skills and Challenges," presented at the International Computing Education Research (ICER 2013), San Diego, CA, USA., 2013.

Proc. of Fourth International Workshop on Cultures of Participation in the Digital Age - CoPDA 2016
Gothenburg (Sweden), October 23, 2016 (published at http://ceur-ws.org).