“This is like a toy, it already got me”: Results of Two Usability Studies of Robotito VPL App with Teachers

ROMINA ABELDAÑO, INCO, Universidad de la República, Uruguay
EWELINA BAKALA, INCO, Universidad de la República, Uruguay
SANTIAGO HITTA, INCO, Universidad de la República, Uruguay
ANA CRISTINA PIRES, Interactive Technologies Institute, Universidade de Lisboa, Portugal

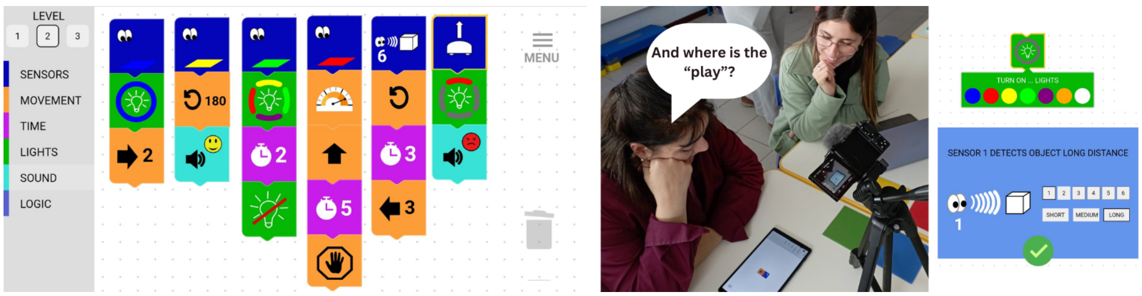

Fig. 1. From left to right: a program built in the newest version of the Robotito App with six behaviors that we asked the teachers to define in the Evaluation 2, a teacher during the evaluation, and two examples of configuration menus.

The relevance of computational thinking (CT) in early education is globally recognized, with Uruguay already incorporating it into kindergarten curricula. However, there is a notable disconnect: existing CT tools, primarily devised for small home groups, fall short in larger classroom settings. To bridge this gap, we developed Robotito VPL – an innovative, free, open-source application tailored to teachers with no programming background. Robotito VPL enables teachers to guide young children in CT learning with Robotito, a robot that engages with its surroundings, supporting a playful, hands-on learning experience for children. This paper presents the results of two usability evaluations with seven teachers. We then translated the outcomes of these evaluations into valuable lessons learned and specific design requirements.

CCS Concepts: • **Human-centered computing** → **User interface programming**; **Empirical studies in ubiquitous and mobile computing**; • **Social and professional topics** → *Computational thinking*.

Additional Key Words and Phrases: Usability, Educators, Mobile Application, Educational robotics

ACM Reference Format:

Romina Abeldaño, Ewelina Bakala, Santiago Hitta, and Ana Cristina Pires. 2024. “This is like a toy, it already got me”: Results of Two Usability Studies of Robotito VPL App with Teachers. In *Interaction Design and Children (IDC ’24), June 17–20, 2024, Delft, Netherlands.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3628516.3659418

1 INTRODUCTION

Computational thinking (CT) has been identified as a critical skill for the 21st century, with its demonstrated benefits for problem-solving abilities, executive functions, and digital literacy [10, 14, 23, 25]. The integration of CT into early education is a global trend, and Uruguay has not been an exception. However, this integration presents both remarkable opportunities and challenges. While robots and applications designed for CT education are typically developed for use in a home setting with small groups and close adult supervision, the classroom environment presents a vastly different context. The scarcity of tools that support the effective use of robotics in educational settings highlights a gap in the current research and practice. To address these challenges, we have developed a free and open-source application, Robotito VPL (VPL stands for Visual Programming Language, a synonym for block-based programming - BBP), which allows users with no programming skills, like educators, children, or researchers, to control and configure an open-source and open-hardware robot named Robotito [21]. Robotito was designed to detect color cards on the floor or objects around it. Robotito has two behaviors [1]. Using colored papers or everyday objects, children control Robotito through playful learning experience with tactile and visual elements. The app we have developed expands Robotito's programming possibilities and allows the users to customize its behaviors and to design a wider variety of activities. Although the app was developed for non-programmers in general, our special focus was placed on educators, as we envision them as the main group of end users and gatekeepers for the integration of technology in education. This paper describes the insights gained in two usuability studies conducted in July-August and December 2023.

2 RELATED WORK

Block-based programming (BBP) has become increasingly prominent in educational settings to teach CT, offering a unique approach to programming that caters to learners of various ages more simplistically and intuitively. BBP employs a visual, puzzle-piece metaphor, making it accessible and engaging for novices in programming. Previous research has shown that this type of approach fosters positive learning experiences and enhances students' logical reasoning, problem-solving skills, creativity, and high-order thinking abilities [5, 17, 20]. By reducing the need for syntactic knowledge, BBP simplifies the programming process for novices and offers a lower entry barrier [22]. There has been a growing development of BBP digital interfaces. However, most apps do not integrate robots into CT activities which offer numerous benefits.

Robots provide a hands-on learning experience, making abstract CT concepts more tangible and engaging for children. This hands-on approach enhances problem-solving, sequencing, decomposition, debugging, action-instruction correspondence, and algorithmic thinking [2, 12]. Many researches underscore the efficacy of educational programmable robots as a valuable tool for introducing CT to preschool children [4, 6, 8, 13, 16]. In this context, there has been an increase in the development of commercial apps designed to control robots for educational purposes, such as *LEGO Mindstorms EV3 Programmer* [11], *Spero Edu* [19], *Blockly for Dash* & Dot [24], and *Roboblocky* [3]. While most of the apps have free versions, the robotics kits themselves are usually purchased separately and can vary widely in price, making incorporation in schools a challenge. Beyond monetary considerations, most of these applications cannot be adapted to specific settings because the code is not open and does not allow any modification. To design solutions tailored for classroom contexts and aligned with unique curricular themes, teachers need adaptable tools. To achieve this goal **participation of teachers in the design process** plays a pivotal role. Teachers can actively participate in the process by evaluating the extent and kinds of tools, influencing the design before its deployment in the classroom

[1]Behavior is a rule or set of rules that connect sensing and acting. For example, if the robot senses a red color patch on the floor, it turns all lights red, moves slowly forward for 3 seconds, and stops

[1, 7, 18]. By recognizing teachers as the gatekeepers and experts who interact daily with children in educational settings, we decided to include them in the design process to develop an app to be integrated in real classrooms.

3 USABILITY EVALUATION

The usability evaluation consisted of multiple phases: a pilot study to delineate the evaluation's scope, Evaluation 1 to pinpoint areas for enhancement, and Evaluation 2 to assess the efficacy of new modifications and identify further areas requiring improvement.

3.1 Robotito VPL

Robotito VPL is an Android application that allows programming behaviors of Robotito [21] using block-based programming. It was developed using the Blockly framework [15] through an iterative design approach with researchers and teachers as possible end-users (more details can be found in [9]). The app evaluated in both usability studies had 20 coding blocks divided into five block categories (see Figure 2).

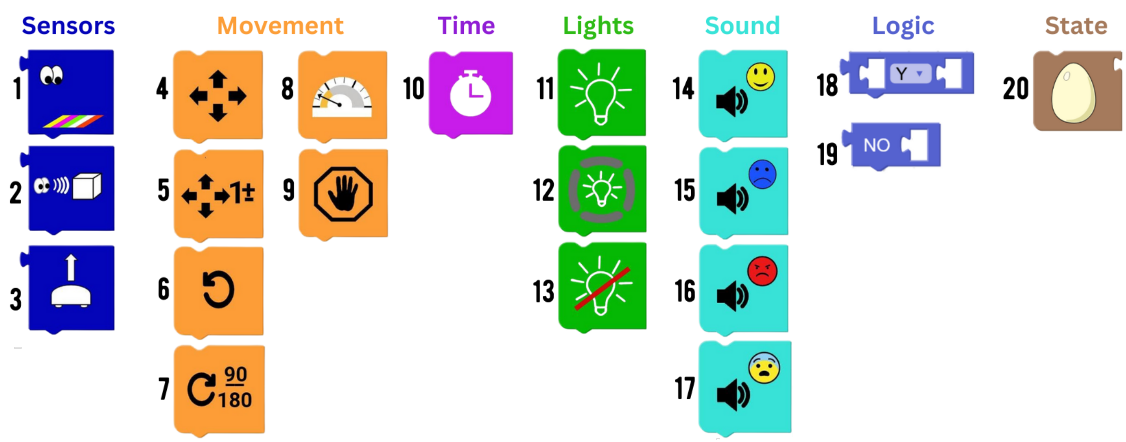

Fig. 2. App design used in Evaluation 1. Block #1 to configure color detection, #2 distance sensor, #3 "robot picked-up" event, #4 move continously in a specific direction, #5 move by steps, #6 spin on spot, #7 rotate by degrees, #8 set velocity, #9 stop, #10 timer, #11 light all leds, #12 light independently zones of LEDs, #13 turn off lights, #14 happy sound, #15 sad, #16 scared, #17 angry, #18 logical and/or, #19 logical negation, and #20 state block.

3.2 Participants and Procedure

The pilot study involved a preschool teacher (T1) with extensive experience in technology who also advises other teachers on integrating technology into their classrooms. Evaluation 1 engaged four teachers who work at the preschool level or with 1st and 2nd grades. Two of them have broad experience (T2, T3) with programming apps, including those for programming robots, while the other two (T4, T5) have no previous experience. Evaluation 2 was conducted with two primary school educators (T6, T7) who had very limited experience with programming apps.

We led the evaluations individually for about 30 to 40 minutes. First, we explained the study's objective and introduced teachers to Robotito. Then, teachers configured six robot behaviors using the app and saved the generated program (see Figure1 for the behaviors specified in Evaluation 2). Tasks were consistent across both evaluations for comparative analysis. We asked the teachers to think aloud during the evaluation and recorded their comments and actions with a camera while capturing the tablet's screen while completing the tasks.

4 RESULTS

We present a summary of the findings from both usability studies.

Visual Differentiation of the Blocks: We observed confusion related to the **function of the blocks** that was not intuitive in some cases. For instance, T2, T4, T6, and T7 confused the "move by steps" with the "move continuously in a specific direction" block. This could be related to the number next to the "move by steps" block, which gave T2: *"an idea of mobility"*. T6 and T7 mistakenly used the number of steps as a measure of time. On the other hand, T3 confused the rotation "spin on spot" block with the "rotate by degrees". Additionally, during various tasks, T3, T4 and T5 had issues in selecting the correct light block between "light independently zones of LEDs" and "light all leds" block. These instances highlight the difficulties in correctly identifying the appropriate blocks due to their similar appearances (see Figure 2).

We also observed that teachers employed **various strategies** to manage the uncertainty regarding the functions of the blocks and to ensure the accuracy of the programs they created. For instance, T4 adopted a **trial-and-error** method with the blocks, adding and removing them to find the most suitable one. On the other hand, T3 and T5 frequently double-checked their choices by consulting **blocks' descriptions**. T3 only used that strategy for the rotation blocks, which shows that she probably understood the other blocks' functions. Interestingly, T3 inquired whether we had a tutorial detailing the functions of the blocks, which was not available at that stage of the project. She emphasized her usual approach: *"Whenever I use something for the first time, I read the manual first."*

Lesson Learned 1: *The pictogram's similarity among related blocks makes distinguishing their functions challenging.*
Design Requirement 1: *Introduce distinct and more intuitive pictograms for similar blocks and a manual highlighting differences within blocks from the same category.*

Organization of the Blocks into Levels: To improve blocks' identification, in Evaluation 2, we decided to reorganize similar blocks into different **levels**. The first level of the "Movement" category only included the "moving by steps" and "rotate by degrees" blocks. In the second level, all the related blocks are presented. However, we observed that organizing blocks into different levels yielded unsatisfactory results. Although this approach did not necessarily made it easier to find the correct block, it could prevent overwhelming teachers with the entire array of blocks during their initial programming learning phase. Consequently, this feature would facilitate a smoother progression for them. As T6 said: *"For me, it's great (...), there are teachers who know [programming] and others who don't, it's like for the whole range [...] having levels is fantastic because [...] the one who knows a bit more can program and have a slightly higher level and the one who can start [...] with the tutorial, it's going to be awesome."*

Lesson Learned 2: *The reorganization of similar blocks into different levels did not improve teachers' identification of the blocks' functions. However, this approach could cater specifically to the needs and programming knowledge of teachers, a beneficial feature acknowledged by educators.*
Design Requirement 2: *Provide various levels that can be selected by teachers, tailored to their specific needs and the activities they have planned.*

Disruptive Informative Pop-ups: In Evaluation 2, we implemented informative pop-ups that provided essential information and usage examples directly within the interface. These pop-ups were designed to activate after a long press on any block. However, we noticed that T6 and T7, while attempting to organize the blocks, often **accidentally**

activated pop-ups. This led to the unintended appearance of pop-ups, which disrupted the teachers' workflow.
Lesson Learned 3: *Teachers inadvertently triggered pop-ups, resulting in unwanted disruptions.*
Design Requirement 3: *Ensure that auxiliary pop-ups do not interfere with normal interaction with the blocks. To minimize unintentional activations, we consider adjusting the duration required to trigger the pop-ups or implement an info-button next to each block to trigger the pop-ups.*

Block Fitting and Alignment: Block fitting and alignment are crucial to creating programs in programming apps. We observed that T4, T5, T6 and T7 faced challenges fitting the blocks with T5, specifically pointing out the absence of clear "**fitting indicators**." To overcome these challenges, teachers adopted various strategies. T7 and T6 initially arranged the blocks from right-to-left in rows while T4 left-to-right. Conversely, T5 adopted a bottom-up approach, starting from the lowermost block. T7 remarked: *"It was challenging because one tends to read from left-to-right and initially thinks that's the way. Then I realized, no, these little parts must be there for a reason, they fit together. That's when [...] I understood the fitting."* However, T6 did not notice the fitting during the activity and had to be instructed at the end. Therefore, she suggested: *"If it were shaped like a puzzle, it would be much easier to detect [...] You look at it and say, oh, it's a puzzle [...]. I'm sure my colleagues will think the same."*
Lesson Learned 4: *Inexperienced teachers struggled to perceive the alignment correctly (top-down) and block fitting as they did not recognize the visual cue indicating where blocks fit together.*
Design Requirement 4: *Include distinctive puzzle-like shapes and increase the size of visual cues of block fitting.*

Color Palette Design and Color Selection: All teachers were able to successfully select colors from color palettes, but they recommended enlarging the squares in the "light independently zones of LEDs" block to avoid incorrect color selection caused by small patches. This feedback has led to a revision and enhancement of the color palette to better meet the visual preferences and needs of the users.
T7 faced difficulties accessing **color palette** in the "light independently zones of leds" block since the palette become available only after selecting a specific color zone. The block configuration pop-up titled "Select a sector" alone did not make it clear to T7 that she needed to press on one of the four zones. She remarked: *"That's not so predictable. It's not clearly identified [...] that's something to improve."*
Lesson Learned 5: *The color palette consisted of small squares, making color selection challenging.*
Design Requirement 5: *Redesign a color palette with larger, circular color swatches.*
Lesson Learned 6: *Accessing color options in the "light independently zones of LEDs" block is not intuitive, and the accompanying text is not entirely clear.*
Design Requirement 6: *Use more intuitive words in the configuration pop-up and preselect one defualt zone to show inmedietly the color palette.*

Block Deletion Process: We observed teachers employing several strategies to **delete blocks**. For instance, while T2 used the "trash can", T3 preferred sliding blocks back into their categories because she found it challenging to drag them accurately into the trash can. T5 tried different strategies as she struggled to delete blocks successfully. Additionally, T2 noted the ability to recover blocks from the trash can, something not commonly seen in other programs, where deletion is usually permanent.
Lesson Learned 7: *The block deletion area was limited and required precise action.*

Design Requirement 7: *Enlarge the trash can area to facilitate easier block deletion.*

Enhanced Auditory Feedback: T2 suggested adding an option to **preview sounds** from sound blocks within the app itself rather than having to execute the program on the robot to hear them.
Lesson Learned 8: *Sound blocks could only be played back through the robot, limiting immediate feedback.*
Design Requirement 8: *Introduce the capability to play sound previews directly from the sound blocks.*

Overlay of Blocks: While exploring the function of different blocks, T6 tended to accumulate and duplicate them on the programming screen. She realized the **duplication** upon moving the time block: *"Here they are double because I put them twice"* and proceeded to delete the duplicates. Overlapping happened because some blocks, if added to the workspace from the menu by double-click (not dragging), appear in the same location.
Lesson Learned 9: *Tendency to accumulate and repeat blocks in the workspace.*
Design Requirement 9: *Improve the visibility and organization of blocks to prevent duplication and overlapping. This could involve ordering blocks in the layout.*

5 DISCUSSION AND CONCLUSION

Our iterative design process was fundamental in gaining insights into teachers' perceptions and usability experiences with the app. The overall response from teachers was positive, even for those teachers who admitted to having limited digital literacy and found the app remarkably easy to use: *"This is like a toy, it already got me"* - T4, *"I like it, it's truly interesting"* - T6, *"I'm not very good at [technology], but I found it super easy. It's great."* - T7. This ease of use might be attributed to the playful, colorful design with simple but appealing drawings and minimal text, which appeared more approachable than formal, text-heavy programming languages. Teachers commented that they found it engaging and even suitable for children: *"[Children] have it clear, apart for them it's trial and error, they have, I'm not saying more patience, but more willingness to use it, [...] it's great."* - T6. However, assessing children's interaction with the app will only be possible through future studies specifically involving them.

We observed intriguing differences in how teachers with and without experience in programming apps approached the app. Experienced teachers instantly examined the app's levels and blocks and tended to search for functionalities akin to those in familiar apps. For instance, they searched for commands like "play" and "end" that stemmed from their experience with programming apps. Additionally, their experience helped them quickly grasp the idea of connecting blocks, unlike inexperienced teachers who struggled to connect blocks, finding it less intuitive.

Usability evaluations have provided valuable insight into how teachers interact with our programming app. Despite the blocks' overall positive appreciation and perceived intuitiveness, these interactions revealed crucial areas that require attention and improvement. The analyses of the evaluation studies identified challenges such as visually highlighting the inset between blocks, achieving clear differentiation between pictograms of blocks with similar actions, and providing additional information about the blocks without interrupting users' workflow. Additionally, teachers suggested incorporating a detailed tutorial with examples to illustrate the programming process. Future updates will consider these aspects to improve the teachers' experience. In conclusion, this study underscores the relevance of involving teachers in creating a programming app to teach CT in classroom settings, providing actional design requirements.

ACKNOWLEDGMENTS

This work was supported by national funds through Agencia Nacional de Investigación e Innovación, Uruguay (project number: FSED-2-2021-1-169697).

6 SELECTION AND PARTICIPATION OF CHILDREN

No children participated in this study.

REFERENCES

[1] Alissa N. Antle. 2013. Research opportunities: Embodied child–computer interaction. *International Journal of Child-Computer Interaction* 1, 1 (2013), 30–36. https://doi.org/10.1016/j.ijcci.2012.08.001

[2] Ewelina Bakala, Anaclara Gerosa, Juan Pablo Hourcade, and Gonzalo Tejera. 2021. Preschool children, robots, and computational thinking: A systematic review. *International Journal of Child-Computer Interaction* 29 (2021), 100337.

[3] Barobo. 2024. Roboblockly. https://roboblocky.com/. https://roboblocky.com/ Accessed: 2024-01-16.

[4] Marina U Bers, Carina González-González, and Mª Belén Armas-Torres. 2019. Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education* 138 (2019), 130–145.

[5] Jen-I Chiu and Mengping Tsuei. 2020. Meta-Analysis of Children's Learning Outcomes in Block-Based Programming Courses. In *HCI International 2020 – Late Breaking Posters*, Constantine Stephanidis, Margherita Antona, and Stavroula Ntoa (Eds.). Springer International Publishing, Cham, 259–266.

[6] Kyriakoula Georgiou and Charoula Angeli. 2019. Developing preschool children's computational thinking with educational robotics: The role of cognitive differences and scaffolding. In *16th International Conference on Cognition and Exploratory Learning in Digital Age, CELDA 2019*. 101–108. https://doi.org/10.33965/celda2019_201911l013

[7] Yiannis Georgiou and Andri Ioannou. 2020. A Co-design Approach for the Development and Classroom Integration of Embodied Learning Apps. In *Learning and Collaboration Technologies. Human and Technology Ecosystems*, Panayiotis Zaphiris and Andri Ioannou (Eds.). Springer International Publishing, Cham, 217–229.

[8] Yen Air Caballero González and Ana García-Valcárcel Muñoz Repiso. 2018. A Robotics-Based Approach to Foster Programming Skills and Computational Thinking: Pilot Experience in the Classroom of Early Childhood Education. In *Proceedings of the Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality* (Salamanca, Spain) *(TEEM'18)*. Association for Computing Machinery, New York, NY, USA, 41–45. https://doi.org/10.1145/3284179.3284188

[9] Santiago Hitta. 2022. Interfaz de configuración de Robotito para usuarios no programadores. (2022).

[10] Sharin Rawhiya Jacob and Mark Warschauer. 2018. Computational thinking and literacy. *Journal of Computer Science Integration* 1, 1 (2018).

[11] LEGO. 2024. LEGO Mindstorms EV3 Programmer. https://education.lego.com/en-us/downloads/mindstorms-ev3/software/. https://education.lego.com/en-us/downloads/mindstorms-ev3/software/ Accessed: 2024-01-16.

[12] Elena Macrides, Ourania Miliou, and Charoula Angeli. 2022. Programming in early childhood education: A systematic review. *International Journal of Child-Computer Interaction* 32 (2022), 100396. https://doi.org/10.1016/j.ijcci.2021.100396

[13] Emily Relkin, Laura E de Ruiter, and Marina Umaschi Bers. 2021. Learning to code and the acquisition of computational thinking by young children. *Computers & education* 169 (2021), 104222.

[14] Judy Robertson, Stuart Gray, Martin Toye, and Josephine Booth. 2020. The relationship between executive functions and computational thinking. *International Journal of Computer Science Education in Schools* 3, 4 (2020), 35–49.

[15] Robobloq. 2024. Blockly. https://developers.google.com/blockly. https://developers.google.com/blockly Accessed: 2024-01-16.

[16] Anika Saxena, Chung Kwan Lo, Khe Foon Hew, and Gary Ka Wai Wong. 2020. Designing Unplugged and Plugged Activities to Cultivate Computational Thinking: An Exploratory Study in Early Childhood Education. *The Asia-Pacific Education Researcher* 29, 1 (2020), 55–66.

[17] Şenol Saygıner and Hakan Tüzün. 2023. The effects of block-based visual and text-based programming training on students' achievement, logical thinking skills, and motivation. *Journal of Computer Assisted Learning* 39, 2 (2023), 644–658.

[18] Stylianos Sergis, Demetrios G. Sampson, María Jesús Rodríguez-Triana, Denis Gillet, Lina Pelliccione, and Ton de Jong. 2019. Using educational data from teaching and learning to inform teachers' reflective educational design in inquiry-based STEM education. *Computers in Human Behavior* 92 (2019), 724–738. https://doi.org/10.1016/j.chb.2017.12.014

[19] Sphero. 2024. Sphero Edu. https://edu.sphero.com/sphero/home. https://edu.sphero.com/sphero/home Accessed: 2024-01-16.

[20] Yu-Sheng Su, Mingming Shao, and Li Zhao. 2022. Effect of mind mapping on creative thinking of children in scratch visual programming education. *Journal of Educational Computing Research* 60, 4 (2022), 906–929.

[21] Gonzalo Tejera, Guillermo Amorin, Andrés Sere, Nicolás Capricho, Pablo Margenat, and Jorge Visca. 2019. Robotito: programming robots from preschool to undergraduate school level. In *2019 19th International Conference on Advanced Robotics (ICAR)*. IEEE, 296–301.

[22] David Weintrop and Uri Wilensky. 2015. To block or not to block, that is the question: students' perceptions of blocks-based programming. In *Proceedings of the 14th international conference on interaction design and children*. 199–208.

[23] Jeannette M Wing. 2006. Computational thinking. *Commun. ACM* 49, 3 (2006), 33–35.

[24] Wonder Workshop. 2024. Blockly for Dash & Dot. https://home.makewonder.com/apps/blockly. https://home.makewonder.com/apps/blockly Accessed: 2024-01-16.

[25] Aman Yadav, Jon Good, Joke Voogt, and Petra Fisser. 2017. Computational thinking as an emerging competence domain. *Competence-based vocational and professional education: Bridging the worlds of work and education* (2017), 1051–1067.