A Multidimensional Dialogic Framework in Support of Collaborative Creativity in Computer Programming

Deller James Ferreira, Federal University of Goiás, Informatics Institute, Campus II Samambaia-Prédio IMFI-Goiânia-Goiás-Brazil-74001-970, deller@inf.ufg.br
Rupert Wegerif, University of Exeter, Graduate School of Education, St. Luke's Campus - Heavitree Road – Exeter-UK - EX1 2LU, R.B.Wegerif@exeter.ac.uk

Abstract: Although in computer programming creativity is required, teaching methods applying tasks that address creative aspects are scarcely reflected in computer science education. This paper describes a multidimensional dialogic framework in support of creativity in programming and the results of a successful computer science experiment, where the teacher facilitates the development of programming skills by means of the students random involvement in collaborative and creative dimensions of the framework. The teacher is called to pay attention in a great repertoire of productive interactions that enables him/her to promote creative programming activities. The framework presented here involves underlying dialogic processes from seven collaborative and creative dimensions that allow students to develop both adaptive and innovative creativity in programming. Students can simultaneously activate two or more ideas, images, or thoughts and have them interact, prompt thought experiments, change cognitive perspectives, raise new points of view, and risk category mistakes during computer programming.

Introduction

Teaching creativity is an important endeavor in computer science education. Although in computer programming creativity is required, teaching methods applying tasks that address creative aspects of programming are scarcely reflected in computer science education research. Despite students should be fostered to develop and apply their creativity, computer science education researchers and teachers does not approach creativity very often. In this work we develop and apply a contextualized dialogic framework to foster creativity in introductory programming courses. During program design, the teacher evokes dialogic processes to trigger exploratory discussions among students. Such students´ discussions have the potential to facilitate the formation of various connections among different programming aspects that propitiate the emergence of creative programmers.

Learning programming demands abilities to combine language features to create computational solutions. Here, we go beyond language features and its combinations. The dialogic pedagogical approach embraces the application of programming patterns and anti-patterns as an important aspect of creative processes during program design. Under this perspective, programming design skills also involves abilities to blend patterns and language features to construct programs. Some studies (Robins et al., 2003; Muller et al., 2004; Soloway & Ehrlich, 1984) show that programming expertise is partly represented by a knowledge base of pattern-like chunks. Programming patterns are are solutions to basic recurring algorithmic problems and form the building blocks for the development of programs.

The use of the pedagogical dialogic framework presented here makes possible exercise both innovative creativity, that is original, transformational, and expressive and adaptive creativity that is logical, adequate, and well-crafted. Programmers can be able to adapt and combine programming patterns, making use of pieces of knowledge well-crafted done by experts, but also be able to create their own new solutions or new patterns for a computational problem blending programming patterns and programming language knowledge development in many creative directions.

Dialogic Approach for Teaching and Learning to be a Creative Programmer

Wegerif (2010) proposes an educational dialogic theory of thinking and teaching thinking, which starts from the metaphor of thinking as dialogue. Dialogic education means teaching for dialogue as well as teaching through dialogue. Under dialogic education perspective, thinking is understood not as some kind of thing but as a kind of relationship that expresses a way of responding to other´s ideas and to new possibilities. He advocates that by teaching students to engage more effectively in complex and meaningful dialogues with others we are teaching them the essence of thinking and proposes a scaffolding approach to teaching thinking as opening, widening and deepening a dialogic space. Students are engaged in a widen dialog space when they are better acquainted with the range of positions that are possible and they deepen a debate when they are able to go deeper into a single bit of the argument to explore its assumptions and implications.

Considering as a starting point Wegerif´s educational dialogic theory, we elaborate in this work a pedagogical dialogic framework for teaching creativity. A dialogic process comprises thinking and discussion,

generating a verbal description of the thinking process. The ideas created underneath dialog processes can be developed in other dialog processes. Dialogic processes explicitly prompt students to be aware of possible ways to create and improve an idea, opening, deepening and widening the dialogic space. We propose the following seven kinds of transformations of dialogic space: immersing, unpacking opportunities, overcoming boundaries, expanding, discovering unpredictable places, and developing.

Dimension 1. Immersing in Programming

Students can widen the design space while discuss having in mind search information having an objective in mind and search information for inspiration, detect relevant and irrelevant information, recognize familiar information and cope with new information, and reapply techniques and adapt techniques.

According to Jonassen (2010), when students scrutinize similar problems for their structures, they gain more robust conceptual knowledge about the problems, constructing stronger problem schema. This dimension concerns with the enhancement of the analogical thinking. Analogical reasoning involves the transfer of solutions from previously known problems to novel ones and the ability to abstract similarities and apply previous productive experiences to new situations. This dimension is also concerned with the search for information. To be successful at discovery and innovation students should be aware of previous and related work and should be aware of principles and techniques to be applied in the development of their work. The more diverse your knowledge, more interesting is the interconnections. In this dimension, given a computational problem, during program design, debug, and re-design, some dialogic processes are: detect programming patterns to be adapted, making connections between problem kind and previous knowledge pieces, adapt patterns to various problem´s constraints, check if minor but essential patterns modifications were not neglected, verifying if not only the main pattern function was considered, i. e., the secondary aspects were not omitted, perform simple mappings of patterns and major patterns´ modifications, approach the problem systematically, and predicting the mixing of patterns, sequence actions, choices and iterations.

Dimension 2. Unpacking Programming Opportunities

Students can deepen the design space when discuss while collaboratively look for attributes and relationships among concepts and new ideas, and try to organize the information, recognize dependence and independence relations, necessary and sufficient conditions, causes and effects, similarities and differences, correspondences and oppositions, class inclusion and exclusion, associations and dissociations, hierarchy ascendant and descendant relations, order and disorder, abstract and concrete features, potential and non-potential uses/functions, examples and counter-examples, and make an interplay between concrete and abstract features.
Guilford (1967) advocates that elaboration and fluency are two fundamental components of the creative process. This dimension embraces the divergent thinking abilities elaboration and fluency. The teacher can boost the students´ improvement of these abilities to explicit what is already there but hidden and also to deal with the who, what, why, and how elements of solution ideas. In this dimension, given a computational problem, during program design, debug, and re-design, some dialogic processes are: establish interconnections among patterns and observe connections between patterns and problems, generalize the examples given by the teacher being able to understand a pattern, avoid specialization problems, in which patterns are maladaptive, and search patterns examples.

Dimension 3. Overcoming Programming Boundaries

This dimension is related to an attempt to overcome and visualize concepts and ideas in an open minded way. This dimension widens the dialog space. The students situate ideas in a bigger context and perform contextual shifting. Jointly search for indirect and direct factors, relationships with "neighbor" ideas outside a given context, visualize scope, limitations, and constraints, in an attempt to overcome it. Seeing an idea in different contexts and also seeing ideas in a bigger scenario is a way to overcome conceptual barriers, gain insight about other possible uses and meanings, recognize indirect and direct factors, bigger and smaller context, limitations and ranges, constraints and lack of constraint, state a context and contextual shifting, and state a point of view and change a point of view. Guilford (1967) advocates that flexibility is a fundamental component of the creative process. This dimension embraces the divergent thinking ability flexibility. In this dimension, given a computational problem, during program design, debug, and re-design, some dialogic processes are: think about consequences of bad use of patterns, approach programming by significant control structures and patterns, trying to develop and the program as a whole or by means of significant parts and search for representative context of patterns usage.

Dimension 4. Expanding Programming

This dimension entangles constructive interactions among students related to innovative construction of a complex system of ideas. This dimension widens the dialogic space. Students make together re-combinations and combinations of similar or distinct of concepts and ideas, try to make combinations of possible disparate or

unconnected ideas by means of a dialectical synthesis or encapsulating the entire dimension of a new concept, derive new knowledge on the basis of a lack of similarity between two or more past constructs or elements from domains which are far apart, make integrations and identify elements, broke and recombine ideas, make usual and unusual interpretation, produce an idea and point the lack of new ideas, and make predictions and findings. Dewey (1929) defined inquiry as a set of operations by which an indeterminate situation is rendered determinate. When participants engage in inquiry together, new meanings are created as a co-production. In this dimension, given a computational problem, during program design, debug, and re-design, some dialogic processes are: develop a repertoire of patterns suitable for adaptation to new formal systems, being aware that patterns emerge from the merging of two or more commands, be aware that programs are combinations of language features and patterns, that language features govern the patterns composition into programs, fit in sequence, nest and merge patterns, check if patterns interactions were not neglected, avoid wrong analogies that emerges from previous experience, given a pattern try to glimpse distinct related aspects like different use contexts, interpret program code, analyze previous patterns interpretations, composition with other patterns, and common misuses.

Dimension 5. Developing Programming

This dimension encompasses the evaluation, critics, and bringing together of ideas. One important aspect of this dimension is that when students evaluate and critique different perspectives and ideas they must be confronted with uncertainty and conceptual conflict. Both are states of disequilibrium that activate a process of conflict resolution and a quest for certainty (Dewey, 1929). Here students deepen and widen the dialogic space. They evaluate, compare, select concepts and ideas, consider different alternatives, and point positive and negative outcomes. They carry out decision making processes in function of criteria application. Here, we consider that a pattern is more efficient than other if it contains less instructions and operations. Given a computational problem, during program design, debug, and re-design, some dialogic processes are: try to improve pattern efficiency, check if the pattern is concise, brief, clear, and legible, compare right patterns and pick the best solution according to efficiency.

Dimension 6. Discovering Unpredictable Places in Programming Solutions

This dimension is related to the exploration of bad ideas and mistakes. The teacher deliberately encourages students to systematically analyze bad and wrong ideas with the aim to develop good ideas from bad ones and learn from mistakes. This dimension capitalizes on often way in which bad ideas become beneficial detours to good ideas. The exploration of good ideas allows a local exploration of the dialogic space, which leaves unexplored large areas of this space (Dix et al., 2006). The exploration of bad ideas pulls the students to new unpredictable places, facilitating a movement to far away places, which thus allows students to overcome the limitation of exploration that good ideas entail. In computer programming context, bad and wrong ideas are recurrent and can be instantiated by programming anti-patterns. In this dimension, the dialogic processes involve programming anti-patterns.

Dimension 7. Exploring Complementary Paths in the Design Space

This dimension involves complementarity. Here, we elaborate dialogic processes based on Ponty´s notion of "chiasm" (Ponty, 1968). In Ponty´s notion of "chiasm," two concepts emerge as complementary ways of referring to an idea. For example, both sides, figure and ground, depend upon each other and can reverse around each other. This divergence is considered to be a necessary and constitutive factor in allowing subjectivity to be possible at all. However, he suggests that rather than involving a simple dualism, the divergence between touching and being touched, or between the sentient and the sensible, mind and body, subject and object, self and other also allows for the possibility of overlapping and encroachment between these two terms. The teacher is called to elaborate the students´ tasks based on complementary concepts.

Assessment of the Dialogic Framework for Programming

For this study, a teacher of an introductory programming course applied traditional course approach in the first part of the course, but asked the students to build programs in groups. In the second part of the course the teacher used the dialogic processes of the framework to prepare programming activities, present content and scaffold students. The teacher presented some program exemplifying programming patterns, presented examples of programming patterns adaptation and combination, provided problems that requires minor and major patterns adaptations and combinations, and motivated students to discuss the solutions that emerged from the group considering dialogic processes during programs planning, debug and evaluation. This mediation process focuses mostly on adaptive creativity in programming. The teacher also presented and analyzed some anti-patterns related to students' misconceptions of students concerning program efficiency. The teacher provided some examples of programs that perform the same tasks comparing their efficiency. Afterwards, the teacher asked for each student in the group to solve new kinds of problems and motivated the group to discuss the partial

solutions and also to discuss the entire innovative solutions in order to choose the best solution during patterns evaluation. This mediation process focuses mostly on innovative creativity in programming.

There were forty undergraduate students from food engineering course. During four months, the data analyzed were programs collaboratively created before and after framework application. The programs were evaluated considering students success in adapting and innovating program codes. First, the observed data was students´ manifestation of language-based anti-patterns, such as sequence, interpretation, inconsistency, and analogy problems. Second, we analyzed pattern problems, i. e., students´ inability to properly combine, specialize, optimize, and visualize details during patterns adaptations. Third, we checked students´ skills to develop solutions to new problems, measured by means of innovative program code generated. Students solved alone and in group 75 problems before framework application and 75 problems after framework application. The experiment results showed that after the teacher application of the dialogic framework language-based anti-patterns and patterns problems decreased while innovative program code generated increased. Statistical overview of the results obtained is shown in Table 1.

Table 1: Statistical Overview of the Results Obtained.

Data after Dialogic Framework Application	Decrease	Increase
Anti-patterns	50,00%	
Patterns problems	42,00%	
Innovative program code		35,00%

Conclusion

The preliminary results indicated that the dialogic dimensions presented here have the potential to allow students collaboratively and creatively explore important problem solving strategies. Students can simultaneously activate two or more ideas, images, or thoughts and have them interact, prompt thought experiments, change cognitive perspectives, raise new points of view, and risk category mistakes. They explore a multi-dimensional space of possibilities of deepening and widening the dialogic space.

References

Dewey, J. (1929). *The Quest for Certainty: A Study of the Relation of Knowledge and Action*. London: George Allen e Unwin.

Dix, A., Ormerod, T., Twidale, B, Michael, B., Sas, C., Silva, A. P. G., and McKnight, L. (2006). *Why Bad Ideas are a Good Idea*, Proceedings of the HCI Educators' Workshop HCIEd.2006-1 Inventivity: Teaching theory, design and innovation in HCI.

Jonassen, D. H. (2010). *Research Issues in Problem Solving*, The 11th International Conference on Education Research New Educational Paradigm for Learning and Instruction.

Guilford, J. P. (1967). *The Nature of Human Intelligence.* New York: McGraw-Hill.

Muller, O., Haberman, B., & Averbuch, H. (2004). (An Almost) Pedagogical Pattern for Pattern-Based Problem-Solving Instruction. In: 2004 Proc. ItiCSE ACM Conf. , 102-106.

Ponty, M. (1968). *The Visible and the Invisible*. Evanston: Northwestern U Press.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and Teaching Programming: A Review and Discussion. *Computer Science Education*, 269-272.

Soloway, E, & Ehrlich, K. (1984). Empirical Studies of Programming Knowledge. *IEEE Transactions on Software Engeneering*, 10(5), 595-609.

Spohrer, J., & Soloway, E. (1986). Novice Mistakes: Are the Folk Wisdoms Correct? *Communications of ACM*, 29(7), 624-632.

Wegerif, R. (2010). The Role of Dialog in Teaching Thinking in Technology. *Dialogue and Development*, 338-357.

Acknowledgments

This work is partially supported by FAPEG and CNPq.