CITRENZ
COMPUTING AND INFORMATION TECHNOLOGY RESEARCH AND EDUCATION NEW ZEALAND

Journal of Applied Computing and Information Technology

ISSN 2230-4398, Volume 19, Issue 1, 2015

Incorporating the NACCQ publications:
Bulletin of Applied Computing and Information Technology, ISSN 1176-4120
Journal of Applied Computing and Information Technology, ISSN 1174-0175

Refereed Poster A1:

Scaffolding, the zone of proximal development, and novice programmers

Awbi, N.K., Whalley, J.L., and Philpott, A. (2015). Scaffolding, the zone of proximal development, and novice programmers. *Journal of Applied Computing and Information Technology*, 19(1). Retrieved January 15, 2015 from http://www.citrenz.ac.nz/jacit/JACIT1901/2015Awbi_Scaffolding.html

Abstract

The work reported here is part of a larger research program that aims to explore the learning strategies that novice computer programmers adopt, the ways in which they integrate knowledge, and the processes they employ when applying their knowledge and skills in different contexts. Our findings, based on a narrative analysis of think-aloud retrospective interviews, indicate that scaffolding can influence progression in learning and can extend a student`s zone of proximal development (Vygotsky, 1978).

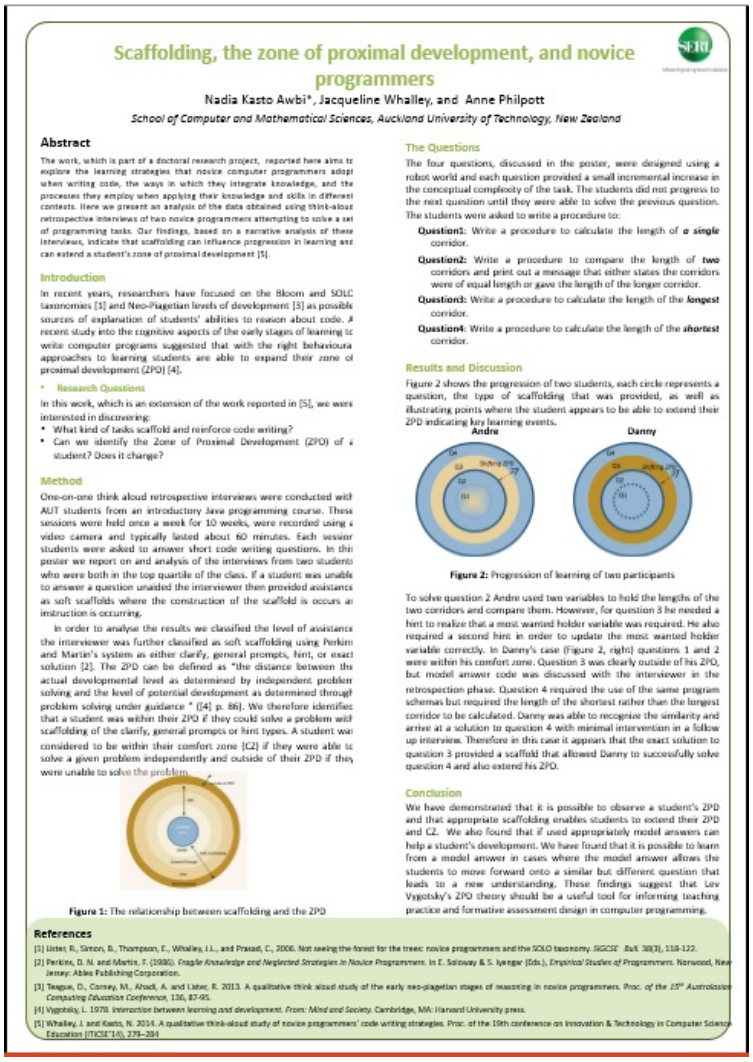
Keywords

K.3 [Computers & Education], Computer & Information Science Education, Computer Science Education

Poster

A3 sized PDF file (289 kb)

A1 sized PDF file (313 kb)



Scaffolding, the zone of proximal development, and novice programmers

Nadia Kasto Awbi*, Jacqueline Whalley, and Anne Philpott

School of Computer and Mathematical Sciences, Auckland University of Technology, New Zealand

Abstract

The work, which is part of a doctoral research project, reported here aims to explore the learning strategies that novice computer programmers adopt when writing code, the ways in which they integrate knowledge, and the processes they employ when applying their knowledge and skills in different contexts. Here we present an analysis of the data obtained using think-aloud retrospective interviews of two novice programmers attempting to solve a set of programming tasks. Our findings, based on a narrative analysis of these interviews, indicate that scaffolding can influence progression in learning and can extend a student's zone of proximal development [5].

Introduction

In recent years, researchers have focused on the Bloom and SOLO taxonomies [1] and Neo-Piagetian levels of development [3] as possible sources of explanation of students' abilities to reason about code. A recent study into the cognitive aspects of the early stages of learning to write computer programs suggested that with the right behavioural approaches to learning students are able to expand their zone of proximal development (ZPD) [4].

Research Questions

In this work, which is an extension of the work reported in [5], we were interested in discovering:

- What kind of tasks scaffold and reinforce code writing?
- Can we identify the Zone of Proximal Development (ZPD) of a student? Does it change?

Method

One-on-one think aloud retrospective interviews were conducted with AUT students from an introductory Java programming course. These sessions were held once a week for 10 weeks, were recorded using a video camera and typically lasted about 60 minutes. Each session students were asked to answer short code writing questions. In this poster we report on and analysis of the interviews from two students who were both in the top quartile of the class. If a student was unable to answer a question unaided the interviewer then provided assistance as soft scaffolds where the construction of the scaffold is occurs as instruction is occurring.

In order to analyse the results we classified the level of assistance the interviewer was further classified as soft scaffolding using Perkins and Martin's system as either clarify, general prompts, hint, or exact solution [2]. The ZPD can be defined as "the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under guidance " ([4] p. 86). We therefore identified that a student was within their ZPD if they could solve a problem with scaffolding of the clarify, general prompts or hint types. A student was considered to be within their comfort zone (C2) if they were able to solve a given problem independently and outside of their ZPD if they were unable to solve the problem.

Figure 1: The relationship between scaffolding and the ZPD

The Questions

The four questions, discussed in the poster, were designed using a robot world and each question provided a small incremental increase in the conceptual complexity of the task. The students did not progress to the next question until they were able to solve the previous question. The students were asked to write a procedure to:

Question1: Write a procedure to calculate the length of ***a single*** corridor.

Question2: Write a procedure to compare the length of ***two*** corridors and print out a message that either states the corridors were of equal length or gave the length of the longer corridor.

Question3: Write a procedure to calculate the length of the ***longest*** corridor.

Question4: Write a procedure to calculate the length of the ***shortest*** corridor.

Results and Discussion

Figure 2 shows the progression of two students, each circle represents a question, the type of scaffolding that was provided, as well as illustrating points where the student appears to be able to extend their ZPD indicating key learning events.

Figure 2: Progression of learning of two participants

To solve question 2 Andre used two variables to hold the lengths of the two corridors and compare them. However, for question 3 he needed a hint to realize that a most wanted holder variable was required. He also required a second hint in order to update the most wanted holder variable correctly. In Danny's case (Figure 2, right) questions 1 and 2 were within his comfort zone. Question 3 was clearly outside of his ZPD, but model answer code was discussed with the interviewer in the retrospection phase. Question 4 required the use of the same program schemas but required the length of the shortest rather than the longest corridor to be calculated. Danny was able to recognize the similarity and arrive at a solution to question 4 with minimal intervention in a follow up interview. Therefore in this case it appears that the exact solution to question 3 provided a scaffold that allowed Danny to successfully solve question 4 and also extend his ZPD.

Conclusion

We have demonstrated that it is possible to observe a student's ZPD and that appropriate scaffolding enables students to extend their ZPD and C2. We also found that if used appropriately model answers can help a student's development. We have found that it is possible to learn from a model answer in cases where the model answer allows the students to move forward onto a similar but different question that leads to a new understanding. These findings suggest that Lev Vygotsky's ZPD theory should be a useful tool for informing teaching practice and formative assessment design in computer programming.

References

[1] Lister, R., Simon, B., Thompson, E., Whalley, J.L., and Prasad, C., 2006. Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. *SIGCSE Bull.* 38(3), 118-122.

[2] Perkins, D. N. and Martin, F. (1986). *Fragile Knowledge and Neglected Strategies in Novice Programmers*. In E. Soloway & S. Iyengar (Eds.), *Empirical Studies of Programmers*. Norwood, New Jersey: Ablex Publishing Corporation.

[3] Teague, D., Corney, M., Ahadi, A. and Lister, R. 2013. A qualitative think aloud study of the early neo-piagetian stages of reasoning in novice programmers. *Proc. of the 15th Australasian Computing Education Conference*, 136, 87-95.

[4] Vygotsky, L. 1978. *Interaction between learning and development. From: Mind and Society*. Cambridge, MA: Harvard University press.

[5] Whalley, J. and Kasto, N. 2014. A qualitative think-aloud study of novice programmers' code writing strategies. Proc. of the 19th conference on Innovation & Technology in Computer Science Education (ITICSE'14), 279–284

1. Introduction

There is no doubt that learning to program is hard and there is a wealth of literature reporting on these difficulties. Difficulty is often attributed to dependency between program concepts (Robins, 2010). Research has found that novice programmers have few schemas available in their long-term memory. Therefore, their knowledge is fragile (Perkins, & Martin, 1986) and the intrinsic cognitive load is high. This high cognitive load (Miller, 1956) means that many novice programmers focus on the programming language syntax and concepts and as a result find the extra load of problem solving impossible. In recent years, researchers have focused on the Bloom and SOLO taxonomies (Lister, Simon, Thompson, Whalley, & Prasad, 2006) and Neo-Piagetian levels of development (Teague, Corney, Ahadi, & Lister, 2013) as possible sources of explanation of students' abilities to reason about code. A recent study into the cognitive aspects of the early stages of learning to write computer programs found that with the right behavioural approaches to learning students are able to expand their zone of proximal development (ZPD) (Whalley, & Kasto, 2014) .

Here we present an analysis of the data obtained using think-aloud retrospective interviews (Van Someren, Barnard, & Sandberg, 1994) of two novice programmers attempting to solve a set of programming tasks. This method is detailed in an earlier paper (Whalley, & Kasto, 2014). The programming tasks were designed to progressively provide for the development of building blocks which make it possible for the student to solve the next problem in the hierarchy of difficulty.

2. The questions

The four questions, discussed in the poster, were designed using a robot world. Each question provided a small incremental increase in the conceptual complexity of the task. For example in order to

solve question two the schema developed in question one, to find the length of a corridor, must be used along with the schema to find the larger of two numbers. The length of the corridor schema requires the use of a plan to count the number of moves a robot makes and one for navigation of the robot within the world. In question 2 there were only ever two corridors. For questions 3 and 4, a correct answer must be able to code with any number (obviously limited by the dimensions of the world) of interconnected n corridors which were always connected at the same point (column 0). The students did not progress to the next question until they were able to solve the previous question. The students were asked to write a procedure to:

1. calculate the length of a single corridor.
2. find the longest corridor of two corridors.
3. calculate the length of the longest corridor.
4. calculate the length of the shortest corridor.

3. Results and discussion

If a student was unable to answer a question unaided the interviewer then provided assistance. In order to analyse the results we classified the level of assistance as either soft or hard (Saye, & Brush, 2002). Soft scaffolding was further classified according to Perkins and Martins system as either *clarify, general prompts, hint, or exact solution* (Perkins, & Martin, 1986). The ZPD can be defined as *"the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under guidance "* ((Vygotsky, 1978) p. 86). We therefore identified that a student was within their ZPD if they could solve a problem with scaffolding of the clarify, general prompts or hint types. A student was considered to be within their comfort zone (CZ) if they were able to solve a given problem independently (Anderson, & Gegg-Harrison, 2013) and outside of their ZPD if they were unable to solve the problem.

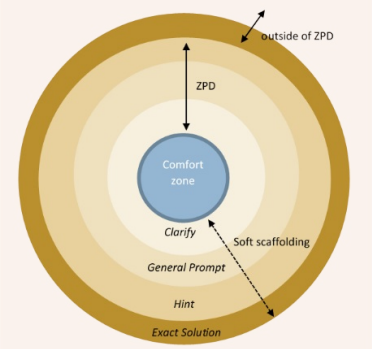


Figure 1: The relationship between scaffolding and the ZPD

Figure 2 shows the progression of two students, each circle represents a question. Colours in the circle indicate the level of assistance (as illustrated in Figure 1) that was provided as well as illustrating points where the student appears to be able to extend their ZPD indicating key learning events for that student and evidence for momentum at the edge of their learning (Robins, 2010).

Figure 2: Progression of learning of two participants

For question 1 Andre required one hint related to program syntax but, while he required assistance, it is likely that question 1 was below or at his actual developmental level. To solve question 2 he used two variables to hold the lengths of the two corridors and compare them. However, for question 3 he realized that this strategy would not work and needed a hint to realize that a most wanted holder variable was required. He also required a second hint in order to update the most wanted holder variable correctly.

In Danny's case (Figure 2, right) questions 1 and 2 were within his comfort zone and are within his ZPD. Question 3 was clearly outside of his ZPD, but model answer code was discussed with the interviewer in the retrospection phase. Question 4 is very similar to question 3. It requires the use of the same program schemas but requires the length of the shortest rather than the longest corridor to be calculated. Danny was able to recognize the similarity and arrive at a solution to question 4 with minimal intervention in a follow up interview. Therefore in this case it appears that the exact solution to question 3 provided a scaffold that allowed Danny to successfully solve question 4 and also extend his ZPD.

4. Conclusion

We have demonstrated that it is possible to observe a student's ZPD and that appropriate scaffolding enables students to extend their ZPD and CZ. We also found that if used appropriately model answers can help a student's development. We have found that it is possible to learn from a model answer in cases where the model answer allows the students to move forward onto a similar but different question that leads to a new understanding. These findings suggest that Lev Vygotsky's ZPD theory should be a useful tool for informing teaching practice and formative assessment design in computer programming.

Acknowledgements

The first version of the poster was displayed and discussed at the First Doctoral Symposium at the 5th annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2014) incorporating the *27th Annual Conference of the National Advisory Committee on Computing Qualifications*, Auckland, New Zealand, October 7-10, 2014.

References

Anderson, N. & Gegg-Harrison, T. 2013. Learning computer science in the comfort zone of proximal development. *Proc of the 44th ACM technical symposium on Computer science education - (SIGCSE '13)*, 495-500.

Lister, R., Simon, B., Thompson, E., Whalley, J.L., & Prasad, C., 2006. Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. *SIGCSE Bull.* 38(3), 118-122.

Miller, G. A. 1956. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review* 63, 2 (1956), 81-97.

Perkins, D. N. & Martin, F. (1986). *Fragile Knowledge and Neglected Strategies in Novice Programmers*. In E. Soloway & S. Iyengar (Eds.), *Empirical Studies of Programmers*. Norwood, New Jersey: Ablex Publishing Corporation.

Robins, A. 2010. Learning edge momentum: a new account of outcomes in CS1. *Computer Science Education* 20, 1 (2010), 37-71.

Saye, J.W., & Brush, T. 2002. Scaffolding critical reasoning about history and social issues in multimedia-supported learning environments. *Educational Technology Research and Development*, 50(3), 77-96.

Teague, D., Corney, M., Ahadi, A. & Lister, R. 2013. A qualitative think aloud study of the early neo-piagetian stages of reasoning in novice programmers. *Proc. of the 15th Australasian Computing Education Conference*, 136, 87-95.

Van Someren, M.W., Barnard, Y.F., & Sandberg, J.A.C. 1994. *The think aloud method a practical guide to modelling cognitive processes*. Academic Press, London.

Vygotsky, L. 1978. *Interaction between learning and development*. From: Mind and Society. Cambridge, MA: Harvard University press.

Whalley, J. & Kasto, N. 2014. A qualitative think-aloud study of novice programmers' code writing strategies. *Proc. of the 19th conference on Innovation & Technology in Computer Science Education (ITiCSE'14)*, 279-284