Scaffolding, the zone of proximal development, and novice programmers


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)

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). Researchers have 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 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 for 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. In order to answer question 1, the student was required to identify the root cause of the problem. For question 2, the student was required to identify the root cause of the problem and then decide on the appropriate action. For question 3, the student was required to identify the root cause of the problem, decide on the appropriate action, and then implement the solution. For question 4, the student was required to identify the root cause of the problem, decide on the appropriate action, implement the solution, and then verify that the solution was correct.

The Questions

1. What is the programming language syntax and concepts that the student is required to understand in order to solve the problem?
2. What is the student’s zone of proximal development (ZPD) and how can it be extended by scaffolding?
3. What is the student’s cognitive load and how can it be reduced by scaffolding?
4. What is the student’s problem-solving strategy and how can it be improved by scaffolding?
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 length of two corridors. 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 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 the 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 analyze the results we classified the level of assistance as either soft or hard (Saye, & Brush, 2002). Soft scaffolding was further classified according to Sutors and Soto (1994) as hint only, general prompts, soft, or assist solution (Sutors, & Soto, 1994). 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 clarifies, 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](image1.png)

Figure 1 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 at risk of not extending their ZPD indicating key learning events for that student and evidence for momentum at the edge of their learning (Robins, 2010).

For question 1, Andre required one hint related to program syntax but, while he required assistance, it is likely that question 1 was below 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 realised that the strategy would not work and needed a hint to realise 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 2. It requires the use of the same program schema 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.

![Figure 2: Progression of learning of two participants](image2.png)

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 at risk of not extending their ZPD indicating key learning events for that student and evidence for momentum at the edge of their learning (Robins, 2010).

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 this poster was displayed and discussed at the First Doctoral Symposium at the 5th Annual Conference of the National Advisory Committee on Computing Qualifications, Auckland, New Zealand, (October 7-10, 2014).

References


Copyright © 2015 Aeub, N.K., Whalley, J.L., and Piikpaatt, A. Journal of Applied Computing and Information Technology (JACIT); ISSN 2230-4298 (Incorporating the Bulletin of Applied Computing and Information Technology, NECIT, ISSN 1786-4120 and Journal of Applied Computing and Information Technology, NACIT; ISSN 1174-0175) Copyright ©2015 CITRENZ.

The author(s) assign to CITRENZ and educational and non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The author(s) also grant a non-exclusive licence to the Journal to publish the content (in print, electronic and other standard media) and in print form within the Journal of Applied Computing and Information Technology. Authors retain their individual intellectual property rights.

Michael Verhaart, Donald Joyce and Nick Wallingford (Eds.). An Open Access Journal. DOI:10.1080/2065057X.2015.1053706, (© authors)