The Use of Wiki in Teaching Programming: Effects Upon Achievement, Attitudes, and Collaborative Programming Behaviors

Yu-Tzu Lin, National Taiwan Normal University, Taipei, Taiwan
Cheng-Chih Wu, National Taiwan Normal University, Taipei, Taiwan
Chiung-Fang Chiu, National Chi Nan University, Puli Township, Taiwan

ABSTRACT

This article explores the feasibility of employing cooperative program editing tools in teaching programming. A quasi-experimental study was conducted, in which the experimental group co-edited the programs with peers using the wiki. The control group co-edited the programs with peers using only the face-to-face approach. The findings show that the co-editing platform was effective in assisting collaborative learning of programming, especially for program implementation. By observing editing histories, students could compare programs and then reflect more deeply about programming. The use of the wiki history tool also helped to illuminate nonlinear and dynamic procedures utilized in programming. Students who engaged more in the collaborative programming or interacted more with partners on the wiki showed greater program implementation achievements. The major benefit of using the wiki was the enhanced ability to observe the dynamic programming procedure and to encounter programming conflicts, which contributed to the process of procedural knowledge acquisition and elaboration.

KEYWORDS

Collaborative Learning, Collaborative Programming, Computer Programming, Co-Writing, Social Learning, Wiki

INTRODUCTION

Computer programming is challenging for most students (Mayer, Dyck, & Vilberg, 1986; Winslow, 1996; Costelloe, 2004). It is a dynamic and non-linear procedure. In traditional programing instruction, teachers usually demonstrate and explain static code without showing the whole non-linear programming procedure (Soloway, 1986). As a result, novice programmers often feel frustrated with the whole process of implementation and, more often than not, cannot finish a whole program by themselves. With the growing interest in computer supported collaborative learning (CSCL), it has been argued that collaborative programming would not only provide students with opportunities for interacting with peers and utilizing each other’s resources and skills but would also facilitate better learning of computer programming (Williams & Jacobs, 2004; Wu, Chang, Shieh, & Lai, 2009; Alorda, Suenaga, & Pons, 2011; Hwang, Shadiev, Wang, & Huang, 2012). Possible reasons why

DOI: 10.4018/IJDET.2018070102

Copyright © 2018, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
collaborative programming can foster problem solving and improve programming skills are many. Firstly, students can foster more ideas from observing others’ programs (Byrne & Lyons, 2001; DeClue, 2003). Secondly, they also gain experience from seeing others’ mistakes (Wei, 2013). Lee (2008) claimed that the metacognitive activities inherited in collaborative programming can foster students’ self-monitoring which can, in turn, improve problem solving abilities. Bravo, Marcelino, Gomes, Esteves, and Mendes, (2005) also showed that paired-collaborative programming activities could help students with problem-solving. Students not only can develop better problem-solving abilities, but also can create more effective programs when using collaborative programming (Bryant, Romero, & Boulay, 2005; Braught, Eby, & Wahls, 2008). Collaborative programming has been shown to help with motivation, engagement, retention, and building confidence in students (McDowell et al., 2006; Chong & Hurlbutt, 2007). Learning tools for collaborative programming must be, therefore, designed specifically and deliberately to support learner engagement (Chong & Hurlbutt, 2007). By browsing the source programs and feedback from peers during beginning stages of learning, students could gradually improve their independent coding skills through imitating peers’ programs or being enlightened by observing peers’ programs. This has been shown to be especially true for low and medium ability students (Hwang, Wang, Huang, & Huang, 2008; Hwang, Shadiev, Wang, & Huang, 2012). Since programming involves declarative, procedural, and strategic knowledge (McGill & Volet, 1997), more adaptive collaborative learning platforms and activities should be developed to help students build different types of knowledge. Traditional programming instruction, however, focuses more on declarative knowledge.

A co-editing platform, such as the wiki, can provide a workspace for “collective creation” and increase the opportunity to observe others’ idea; therefore, it is frequently used as a collaborative learning tool (Wheeler, Yeomans, & Wheeler, 2008; Moskaliuk, Kimmerle, & Cress, 2009). “Version control” of wikis has been shown to be an important feature for student learning. Students can keep track of all edits and compare the difference between two successive versions, which increases opportunities for reflection (Naismith, Lee, & Pilkington 2011). The utilization of this technique might also provide opportunities for observing the dynamic procedure of programing. As such, there is a great deal of research which supports the efficacy of the co-editing platform in education (Wheeler, Yeomans, & Wheeler, 2008; Moskaliuk, Kimmerle, & Cress, 2009). There is, however, a paucity of studies investigating its efficiency for programing learning in peer scaffolding (Kim & Hannafin, 2011; Hwang et al., 2012). One such study has documented the need for further development of revealing the dynamic procedure of programing (Soloway, 1986). Such studies have cited the need for the clear learning guidelines as well as identifying stages of learning which might be targeted for implementation of co-editing, interactive techniques for student learning (Hwang et al., 2008).

Based on Piaget’s theory on cognitive development (Piaget, 1963, 1970), the most critical factor in a learner’s cognitive development is interaction with peers. Interaction provides opportunities for the learner to compare old ideas with new information from peers. This often takes the form of an intellectual disagreement or argument, which creates cognitive conflict. To solve cognitive conflicts, assimilation and accommodation of knowledge is required. Cress and Kimmerle (2008) developed a model to understand how users assimilate and accommodate their knowledge on the wiki based on Piaget’s theory. On the wiki, divergence of ideas produced incongruity between prior knowledge and new information (Puntambekar, 2006). Although many studies regarded the wiki as an effective environment that supports the collaborative knowledge building process, one study by Cole (2009) reported a lack of interest from students in wiki-based learning. This should remind researchers and instructors that a rigorous instructional design is essential if wiki-based activities are to be successful. Instead of interacting with peers randomly, scaffolding should be provided when using computer supported collaborative learning (Chao & Lo, 2011).

An Organizational Memory Tool for E-Learning
www.igi-global.com/chapter/organizational-memory-tool-learning/27509?camid=4v1a