Using a Multi-Touch Table to Develop Collaborative Teaching Programming Resources in the New Computing Curriculum

Ahmed Alzahrani
Leeds University
scaaal@leeds.ac.uk

Eric Atwell
Leeds University
e.s.atwell@leeds.ac.uk

Andy Bulpitt
Leeds University
a.j.bulpitt@leeds.ac.uk

Abstract

Information and communication technology (ICT) has been questioned extensively throughout recent years and is often considered unsatisfactory and antiquated (Society, 2012). However, a new computing curriculum will be incorporated into school systems in order to replace existing frameworks. This new paradigm focuses heavily on programing and critical thinking, largely at the primary level of schooling. The existing ICT curriculum was considered insufficient and unsatisfactory, consequently highlighting the need for change. Nonetheless, basic ICT applications need to be considered when producing new curriculum in order to expand on familiar principles that students utilize (Gov.uk, 2013).

This project aims to develop a new resource to enhance collaborative programming in UK schools for pupils studying at Key Stage 3 level, i.e., 11–14 year olds. The project started with a review and evaluation of the available tools for collaborative pair programming, such as Scratch Programming (Scratch, 2014), in order to determine their suitability for use in multi-touch tables. Then, the tool Turtle Art (Turtle Art, 2014) was selected to receive further development to be run on a multi-touch table. In the final stage, the usability of the new software, in terms of collaborative pair programming, will be tested and evaluated against the project goals.

The new resource software is intended to be used as an environment for collaborative learning and pair programming, and a number of studies have proven that these educational methods provide an improved quality of education within the curriculum of school systems (see, for example, Wilson et al., 1993; Yerion and Rinehart, 1995; and McKinney and

Denton, 2006 for collaborative learning and Nosek, 1998; Williams et al., 2000a, 2000b; Cockburn et al., 2000; Williams et al., 2001; Mcdowell et al., 2002; Gallis et al., 2003; Cao and Xu, 2005; Brereton et al., 2009; Goel et al., 2010; Rick et al., 2011; and Harlow and Leak, 2014 for pair programming). Thus, the proposed resource software can be beneficial when used for collaborative learning and pair programming for many reasons such as the following:

- It helps in collective decision-making and problem-solving, which leads to enhanced learning and development.
- It focuses on the development of skills that are necessary for employment in contemporary industries.
- It is more enjoyable for developers, increases work satisfaction, and promotes student learning.
- It leads to better student activity engagement.
- It contributes to enhancing the level of student understanding with regard to programming modules.
- It makes students more confident in their work as well as more positive about programming.
- It encourages the development of problem-solving aptitudes.
- It provides more efficiency and quality in understanding the processes that one needs to go through in programing to solve problems.
- It helps students create new ideas and ways to interact with new technologies.
- It encourages participants to practice problem-solving skills in practical circumstances.

References

Brereton, P., Turner, M. and Kaur, R. (2009). Pair programming as a teaching tool: a student review of empirical studies. pp.240--247.

Cao, L. and Xu, P. (2005). Activity patterns of pair programming. pp.88--88.

Cockburn, A. and Williams, L. (2000). The costs and benefits of pair programming. Extreme programming examined, pp.223--247.

Gallis, H., Arisholm, E. and Dyba, T. (2003). An initial framework for research on pair programming. pp.132--142.

Goel, S. and Kathuria, V. (2010). A novel approach for collaborative pair programming. Journal of Information Technology Education: Research, 9(1), pp.183--196.

Goel, S. and Kathuria, V. (2010). A novel approach for collaborative pair programming. Journal of Information Technology Education: Research, 9(1), pp.183--196.

Gov.uk, (2013). National curriculum in England: computing programmes of study - PublicationsGOV.UK.[online]Availableat:https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study[Accessed15Jun.2014].

Harlow, D. and Leak, A. (2014). Mapping studentsâ€™ ideas to understand learning in a collaborative programming environment. Computer Science Education, 24(2-3), pp.229--247.

McDowell, C., Werner, L., Bullock, H. and Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. 34(1), pp.38--42.

McKinney, D. and Denton, L. (2006). Developing collaborative skills early in the CS curriculum in a laboratory environment. 38(1), pp.138--142.

Nosek, J. (1998). The case for collaborative programming. Communications of the ACM, 41(3), pp.105--108.

Rick, J., Marshall, P. and Yuill, N. (2011). Beyond one-size-fits-all: How interactive tabletops support collaborative learning. pp.109--117.

Scratch.mit.edu, (2014). Scratch - Imagine, Program, Share. [online] Available at: http://scratch.mit.edu/ [Accessed 9 Oct. 2014].

Society, B. (2012). Shut down or restart?The way forward for computing in UK schools. The British royal society, [online] The Royal Socity(UK), pp.1-34. Available at: https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf [Accessed 12 Jan. 2012].

Turtle Art, (2014). Activities/Turtle Art - Sugar Labs. [online] Wiki.sugarlabs.org. Available at: http://wiki.sugarlabs.org/go/Activities/Turtle_Art [Accessed 24 Oct. 2014].

Williams, L., Kessler, R., Cunningham, W. and Jeffries, R. (2000). Strengthening the case for pair programming. IEEE software, 17(4), pp.19--25.

Williams, L. and Kessler, R. (2000). All I really need to know about pair programming I learned in kindergarten. Communications of the ACM, 43(5), pp.108--114.

Williams, L. (2001). Integrating pair programming into a software development process. pp.27--36.

Wilson, J., Hoskin, N. and Nosek, J. (1993). The benefits of collaboration for student programmers. ACM SIGCSE Bulletin, 25(1), pp.160-164.

Yerion, K. and Rinehart, J. (1995). Guidelines for collaborative learning in computer science. ACM SIGCSE Bulletin, 27(4), pp.29-34.