Computers & Education 57 (2011) 1202–1213

Contents lists available at ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Modelling programming performance: Beyond the influence of learner characteristics

Wilfred W.F. Lau*, Allan H.K. Yuen

Faculty of Education, The University of Hong Kong, Pokfulam Road, Hong Kong

A R T I C L E I N F O

Article history:
Received 21 April 2010
Received in revised form
6 November 2010
Accepted 6 January 2011

Keywords:
Improving classroom teaching
Pedagogical issues
Programming and programming languages
Secondary education
Teaching/learning strategies

A B S T R A C T

In the 21st century, the ubiquitous nature of technology today is evident and to a large extent, most of us benefit from the modern convenience brought about by technology. Yet to be technology literate, it is argued that learning to program still plays an important role. One area of research in programming concerns the identification of predictors of programming success. Previous studies have identified a number of predictors. This study examined the effect of a combination of predictors (gender, learning styles, mental models, prior composite academic ability, and medium of instruction) on programming performance. Data were collected anonymously through a website from 131 secondary school students in Hong Kong who opted for computer programming in the Computer and Information Technology curriculum. Partial Least Squares (PLS) modelling was used to test a hypothesized theoretical structural model. All of the five aforementioned variables were either direct or indirect predictors of programming performance and the antecedents accounted for 43.6% of the variance in programming performance. While this study shows the influence of learner characteristics such as gender, learning styles, and mental models on programming performance, it highlights the effect that prior composite academic ability and medium of instruction exert on learning outcomes, which is uncommon among studies of similar purpose. These findings have significant implications for policy makers and educators alike.

1. Introduction

Undoubtedly, we are living in a high technology society. Technology products become more compact in size and yet versatile in functions. Oftentimes, they have appealing looks and possess certain degree of mobility and portability. In the Internet world, social networking tools such as Facebook and Twitter have emerged and transformed the ways we live and learn. Although learning to program appears to lose its popularity, we argue that it is still one of the critical technology literacy skills for students today. A document by the United Nations Educational, Scientific and Cultural Organization (UNESCO) depicted four stages of Information and Communication Technology (ICT) use in schools, namely ICT Literacy, Application of ICT in Subject Areas, Infusing ICT across the Curriculum, and ICT specialization, mastery of basic and advanced programming techniques was identified as being instrumental to attaining the level of competence in the last stage (Khvilon & Patru, 2002). More recently, Cohen and Haberman (2007) contended that in this information era, Computer Science is an avenue for us to be technology literate and the authors described Computer Science as a language of technology. As a core component of Computer Science, learning to program still plays an important role in promoting technology literacy. Jones (1995) discussed the debate over whether university training for Computer Science students should be more oriented towards theory than practice and suggested that more effort is required to prepare graduates for typical software job experience. In other words, it can be said that programming education can foster technology literacy in schools and provide training for employees in the workplace. Given the importance of programming education, it is no wonder that programming is an essential topic for K-12 and introductory Computer Science curricula (Kordaki, 2010).

Research into Computer Science education has spanned a broad range of topics in the last four decades (Pears, Seidman, Eney, Kinnunen, & Malmi, 2005). One particular area that has been widely studied is the investigation of predictors of programming success in terms of course performance. Over the years, numerous models have been developed to predict programming performance. Subramanian and Joshi

* Corresponding author. Tel.: +852 28592540; fax: +852 25406360.
E-mail address: wilfredlau@graduate.hku.hk (W.W.F. Lau).

0360-1315/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2011.01.002

(1996) illustrated that two generic computer aptitudes and the ability to find similarities between dissimilar items and to detect internal order in a number and/or letter sequence significantly predicted programming performance. Charlton and Birkett (1999) demonstrated that gender, personality, intellect and computer attitudes, ownership, and experience were predictive of computing behaviour and course performance. Wilson (2002) found five predictive factors for success in a computer programming course, namely comfort level, math background, attribution to luck, a formal class in programming, and game playing. Jones and Burnett (2008) showed that spatial ability, as measured by a mental rotation test, was positively correlated with performance in programming modules. Wiedenbeck and her colleagues (Ramalingam, LaBelle, & Wiedenbeck, 2004; Wiedenbeck, 2005; Wiedenbeck, LaBelle, & Kain, 2004; Wiedenbeck, Sun, & Chintakovid, 2007) investigated a number of models for predicting course performance of novice programmers. These studies identified previous programming experience, self-efficacy, knowledge organization, mental models, computer interest, and computer playfulness as significant predictors of programming performance, either directly or indirectly.

Recently, there is a trend towards studies of predictors of programming success that are robust across nations and/or institutions (Bergin & Reilly, 2006; de Raadt et al., 2005; Simon, Cutts, et al., 2006; Simon, Fincher, et al., 2006). Admittedly, there are a multitude of variables influencing programming performance. This study examined the effect of a combination of predictors (gender, learning styles, mental models, prior composite academic ability, and medium of instruction) on programming performance because previous studies have shown the effect of gender on programming performance (Byrne & Lyons, 2001; Crews & Butterfield, 2003; Pillay & Jugoo, 2005), the effect of learning styles on programming performance (Davidson, Savenye, & Orr, 1992; Drysdale, Ross, & Schulz, 2001; Ross, Drysdale, & Schulz, 2001), and the effect of mental models on programming performance (Acton, Johnson, & Goldsmith, 1994; Cooke & Schvaneveldt, 1988; Trumpower & Goldsmith, 2004) but research on their combined effect on programming performance remains scant. Further, this study also incorporated prior composite academic ability and medium of instruction (MOI) as potential predictors. Prior composite academic ability of a student is believed to affect programming performance and it is expressed in terms of a band in this study. The band of a student was evaluated based on his or her academic performance in Chinese, English, and Mathematics in Primary 5 (Grade 5) and Primary 6 (Grade 6) to be scaled by external assessment. The term MOI in this study refers to the language used for teaching purpose and is widely used in the research community. Research indicates that the MOI does have an effect on programming performance (Lau & Yuen, 2010b). It is noteworthy that a fine-tuning of the language policy has come into effect from the 2010/11 academic year onwards so that the MOI for junior secondary students is applied at the class instead of school level (Hong Kong SAR Government, 2010).

This study differs from previous studies in a number of ways. First, it examined the inter-relationships among the variables in an integrative model. Second, the model presented here intended to incorporate variables at the policy and learner levels, i.e. band and MOI at the policy level and gender, learning styles, and mental models at the learner level, and thus extended previous work in related research which primarily considered the influence of learner characteristics only (Charlton & Birkett, 1999; Ramalingam et al., 2004). This study is contextualized in Hong Kong because many related studies were performed in the 1980s or 1990s with few considerations of learner characteristics (Au, 1992; Chung, 1988; Kong & Chung, 1989). Despite this, it is believed that this study can provide insights into research on students who are likely to be impacted by similar educational policies. In sum, this study attempted to test a parsimonious model that includes important variables identified in the Computer Science education literature and explains as much variance in programming performance as possible. The model clearly has its limitations in that it cannot provide the depth of understanding that individual variables can reveal. However, it satisfies the pragmatic need of policy makers and educators for one which can promote communication, research, and support.

This article is organized as follows. First, a literature review on the effect of the aforementioned predictors on programming performance is presented. The learning style instrument (Gregorc Style Delineator (GSD)) and mental model assessment method (Pathfinder Scaling Algorithm (PSA)) with relation to programming performance are discussed. Second, the methodology used in this study is delineated and in particular, the Partial Least Squares (PLS) modelling approach is described. This is followed by the results and discussion sections. Finally, a conclusion summarizing the major findings in this study is provided.

2. Literature review

2.1. *Effect of learning styles on programming performance*

There is a proliferation of learning style instruments available in the academic literature and in this study, the GSD (Gregorc, 1984) was selected because there is evidence that certain styles predicted by the instrument influence programming performance. The instrument classifies an individual into either a concrete sequential (CS), abstract sequential (AS), concrete random (CR), or abstract random (AR) learner. CS learners tend to perceive reality through their physical senses and think in an orderly, logical, and sequentially manner. AS learners are logical and analytical individuals who have a preference for mentally stimulating task and environment. CR learners like to experiment with ideas and concepts and think intuitively, instinctively, impulsively, and independently. AR learners have a strong sense on the world of feeling and emotion and tend to think in a non-linear and emotional manner.

In a longitudinal study of 4546 students over a period of 4 years (1993–1997), Drysdale et al. (2001) found that in a Computer Science course ($n = 804$), dominant CS learners outperformed their counterparts of other learning styles. In particular, 23% of CS learners received A grades. The corresponding percentage for AS, CR, and AR learners was 20%, 10%, and 7% respectively. When comparing the mean GPAs of different learning style groups, CS learners obtained a mean GPA of 2.98. This was followed by AS learners with a mean GPA of 2.83. CR learners had a mean GPA of 2.50 whereas AR learners obtained a significantly lower mean GPA of 2.12. Drysdale et al. further interpreted that random learners may “struggle in these courses (science and math courses) because of the structure involved in the class and lab components” (p. 285). Although computer programming was not explicitly mentioned in the course content, some parts of the course like computer system fundamentals, computer communication, database management, and spreadsheet may likely require knowledge of programming in order to understand the concepts.

In another study, Ross et al. (2001) investigated the effect of learning styles on academic performance of two computer applications courses. In one course offered to pre-service teachers ($n = 168$), some programming (using HyperCard and HTML) was required. A significant learning style group effect on learning outcomes was observed. Dominant AS learners achieved the highest group score (mean GPA = 3.72).