Computers & Education 56 (2011) 527–538

Contents lists available at ScienceDirect

Computers & Education

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

Empowering teachers to create educational software: A constructivist approach utilizing Etoys, pair programming and cognitive apprenticeship

Young-Jin Lee*

The University of Kansas, 1122 West Campus Road, Room 413, Lawrence, KS 66045, USA

ARTICLE INFO

Article history:
Received 31 May 2010
Received in revised form
22 September 2010
Accepted 25 September 2010

Keywords:
Interactive learning environment
Media in education
Programming and programming languages
Teaching/learning strategies

ABSTRACT

This study investigates whether a visual programming environment called Etoys could enable teachers to create software applications meeting their own instructional needs. Twenty-four teachers who participated in the study successfully developed their own educational computer programs in the educational technology course employing cognitive apprenticeship and pair programming approaches as the primary instructional strategies. Two educational software programs created by the participating teachers were described in order to explain what they were trying to do using Etoys and how they accomplished their goals. The results of an anonymous survey evaluating the difficulty of and the attitude toward learning Etoys indicate that teachers enjoyed learning Etoys and would like to continue to use it in the future although they found it was slightly more difficult, compared to their self-evaluated computer skill. The strengths and weaknesses of Etoys, the difficult computer programming concepts, and the educational implications of Etoys programming were also discussed.

1. Introduction

With the rapid development in technology, computers have become an integral part of our society, and education is not an exception. According to the recent reports from the National Center for Education Statistics (NCES), 91% of K-12 school students have access to computers (DeBell & Chapman, 2006), and the student-to-computer ratio in public schools is 3.8 (Snyder & Dillow, 2010). Despite the significant increase in the number of computers available in schools, however, the instructional methods adopted in K-12 schools do not take full advantage of the latest computer technologies; the most common use of computers in K-12 education is word-processing and the computers are not tightly integrated into the curriculum (Collins & Halverson, 2009; Cuban, 2001).

One of the reasons for the under use of computers in K-12 schools, among others, is the lack of educational software that meets teachers' need in the classroom. Developing educational software is quite different from developing non-educational software, such as business applications, because it should be designed to facilitate the "learning" of its users who may not possess the knowledge under study, rather than improve the "productivity" of its users who usually have enough background knowledge required to perform the given task (Soloway, Guzdial, & Hay, 1994). Software engineers, who may not have a clear understanding of how learning processes could be supported and facilitated, tend to create an inflexible, stand-alone product that is hard to integrate into the processes and methods that teachers would like to employ to help their students learn. Saettler (2004) notes that educational technology products created by non-educators often emphasize an efficient transmission of content knowledge instead of specific learning by individual students. He also points out that educational software should be flexible in order to accommodate a great variety of students who may have different amounts of background knowledge and need different styles of learning approaches. Thus, developing effective educational software requires not only technical knowledge of computer programming, but also thorough understanding of pedagogical content knowledge which allows educators to represent and formulate subject matter knowledge in ways that learners can easily comprehend (Shulman, 1986). The dilemma is that software engineers who have technical knowledge of computer programming usually do not possess enough pedagogical content knowledge while teachers who have expertise in pedagogical content knowledge normally do not know about software development processes.

* Tel.: +1 785 864 0625; fax: +1 785 864 4697.
E-mail address: yjlee@ku.edu.

0360-1315/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2010.09.018

The primary objective of this study is to investigate whether an innovative educational technology tool called Etoys (http://www.squeakland.org/) could allow teachers who have no prior computer programming experience to develop educational software applications specifically designed for their own instructional needs. Also, this study examines how difficult it is for teachers to learn Etoys programming and their attitudes toward developing their own educational software applications using Etoys.

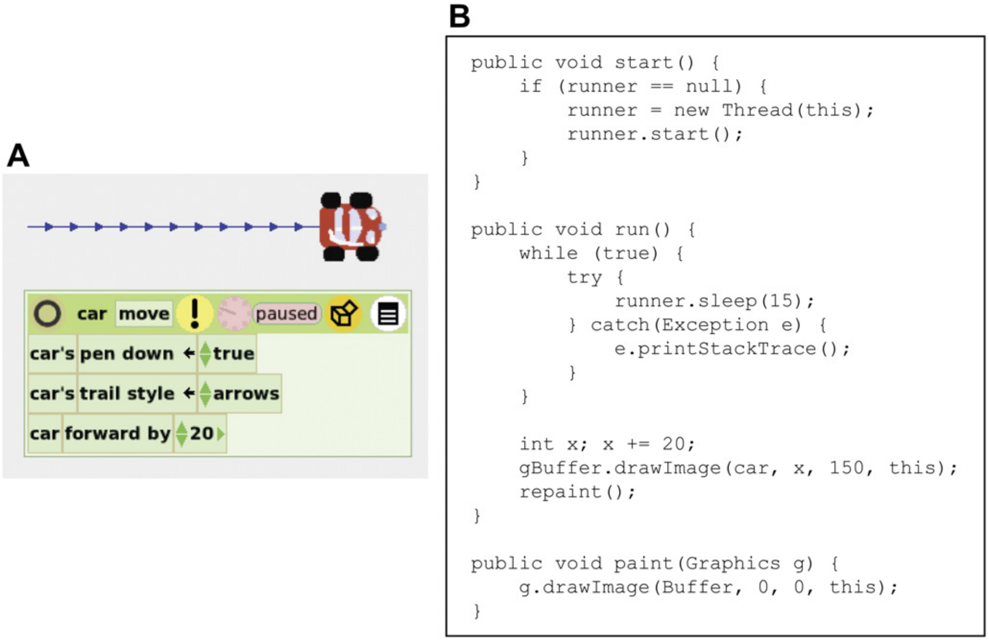
2. Etoys: a visual programming environment for teachers and young children

Etoys is a visual programming environment designed especially for teachers and young children who do not have prior experience in computer programming. Developing software in Etoys starts with creating a two-dimensional graphic object called a Sketch by either using a built-in paint tool available in Etoys or importing an external image file. Once a Sketch is created, its behavior (e.g., movement, appearance or sound) can then be controlled by attaching what is called a Script to it. Unlike conventional programming languages, such as C++ or Java, which requires composition of syntactically correct programming codes in text, Etoys allows its users to create Scripts by putting together visual programming tiles. Since the visual programming tiles in Etoys can be assembled only when they are syntactically correct, they practically remove the need for debugging because assembled programming tiles are guaranteed to conform to correct Etoys programming language syntax. Fig. 1 compares the visual programming approach employed in Etoys to the conventional approach in a text-based programming language.

Another strength of Etoys is that it is designed to develop multimedia products such as digital storybooks, animations or games. Since it provides several programming tiles that can be used to handle graphic images and sound clips out of the box, even complete novices can easily create their first computer animation program incorporating sound effects within an hour or so. Moreover, Etoys comes with many built-in widgets, such as Book and Playfield, that can be readily used in creating educational materials. Education research using Etoys is still in its infancy and only handful of preliminary research studies have been conducted so far. For instance, some researchers used Etoys in developing problem-based learning curriculums for information science and global environmental education (Fujioka, Takada, & Hajime, 2006; Matsuoka et al., 2007; Valente & Osório, 2008). Also, it has been adopted by the One Laptop per Child (OLPC) project as one of the core educational software applications installed on their "One Hundred Dollar Laptop" ("Etoys, 2010"). Most recently, Tagliarini, Narayan, and Morge (2010) started conducting a research project trying to infuse information technology skills into the Science, Technology, Engineering, and Mathematics (STEM) curriculums in the middle and high school levels using Etoys. Clearly, more research effort is needed in order to better understand how Etoys can be used to enhance the way we teach and learn with computers in K-12 schools.

3. Context of research and participants

The primary objective of this study is to investigate whether Etoys could enable educators who have no prior experience in computer programming to develop educational software applications meeting their own instructional needs. In order to achieve this goal, an exploratory case study was conducted with twenty-four students taking an educational technology course offered at the University of Kansas in fall 2008 and 2009 semesters. Of twenty-four students who participated in the study, there were four pre-service teachers, ten in-service teachers and ten teacher educators who were either faculty members in the teacher education department/program in a higher education institution or university staff responsible for professional development for K-12 teachers. Prior to this course, students have taken an introductory educational technology course focusing on developing education media, such as Web sites, movies and podcasts, using



```
car move
car's pen down ← true
car's trail style ← arrows
car forward by 20
```

```
public void start() {
    if (runner == null) {
        runner = new Thread(this);
        runner.start();
    }
}

public void run() {
    while (true) {
        try {
            runner.sleep(15);
        } catch(Exception e) {
            e.printStackTrace();
        }
    }

    int x; x += 20;
    gBuffer.drawImage(car, x, 150, this);
    repaint();
}

public void paint(Graphics g) {
    g.drawImage(Buffer, 0, 0, this);
}
```

Fig. 1. (A) An example Etoys Script moving a car Sketch while leaving arrow-shaped trails; (B) A Java program performing a similar, but simpler, task.