Chapter 7

Analogical Thinking Based Instruction Method in IT Professional Education

Tokuro Matsuo
Yamagata University, Japan

Takayuki Fujimoto
Toyo University, Japan

ABSTRACT

In designing a new teaching system, a challenging issue is how the system intelligently supports learners. This paper describes a methodology and a system design on the intelligent instruction support for software engineering education. For information science courses at a university, software engineering subjects are usually compulsory and students study dominant conceptions of implementation like software architecture, and the methodology of software design in software engineering lectures. To enhance learners’ understanding, the authors design a novel instructional model based on the analogical thinking theory. The analogical thinking-based instruction consists of concrete teaching methods like analogy dropping method, self role-play method, and the anthropomorphic thinking method. Questionnaires for learners after the instructions give results of effective education in an actual trial. The contribution of this paper is to provide a new instruction theory, the way of educational practice method, and implementation of the system.

INTRODUCTION

As one field of engineer education, software engineering instruction method is proposed by many literatures (e.g., Casado-Lumbreras et al., 2009; Chao et al., 1992; Colomo Palacios et al., 2010; Dewayne et al., 2000; Saiedian et al., 2002; Shaw, 2002). When it is designed as an intelligent instruction system, system designers consider how the system supports users by taking students’ activities and the cognitive analyses into consideration (e.g., Baker, 2007; Garcia-Crespo et al., 2008a; Martin, 1998; Miyake & Masukawa, 2000). In this paper, we discuss the support process and learning process to help learners understand and grasp software engineering issues. Such processes can
be implemented into the education support system and intelligent instruction agent in the system.

Nowadays, Universities and other Training Institutions need to clearly identify the Information Technology skills that companies demand from practitioners, including software engineers (Trigo et al., 2010). Universities and educational institutes provide lectures and exercises of software engineering for students who want to take them. In the lecture and exercises, the learning issues include software design, lifecycle models and its history, UML, and so on (e.g., Garcia-Crespo et al., 2008b; Garcia-Crespo et al., 2009b). Learning issues also include the aspect of business model and usability. Students can appropriately understand them by imagining, since many students learn such theories for the first time when they attend university lectures. In many engineering education, students can sometimes visually understand through experiments and exercises (e.g., Behling et al., 1996; Colomo-Palacios et al., 2008; Garcia-Crespo et al., 2009a). However, it is difficult for many students to grasp and design software, since software is not visual. Although the programming exercises in the lectures help learners implement the applications, such applications are quite small and do not have any requirement of industry.

Many students have not studied software engineering when they were a junior high school and high school students. Software engineering has little relationship with the learning subjects in secondary schools. If students have studied subjects such as mathematics, the students would not have many problems with lectures such as information mathematics. On the other hand, there are few students who have learnt about planning and designing software and applications in during secondary school. Law schools and business schools in universities often provide discussion-based lectures since many students are unfamiliar with such learning fields before their entrance into the university. Similarly, software engineering education should also be done based on a method in which students can learn easily.

In this paper, we propose a new software engineering education methodology based on analogical thinking to develop and implement as intelligent tutoring system. Concretely, the analogical thinking-based teaching method consists of the instruction method based on analogical dropping, the instruction method based on self role-play, and the instruction method based on anthropomorphic thinking. We also, in this paper, provide an evaluation of our proposed instruction methodology with data from our experiments. Our research contributes to providing a new instruction theory for software engineering, the way of educational practice based on the new instruction method, and implementation of the tutoring system.

When people understand a certain issue, they sometimes learn it through experience, pre-knowledge, and other stimulus in the environment and course curricula. To develop an instruction system that has such learning process, this paper discusses the methodology of intelligent instruction based on formalization of analogical thinking-based learning model. In our learning model, users’ knowledge is enhanced by their pre-knowledge, developmental theme, developmental stage theory. Each issue is mapped as a computer system and device through integration and reuse with each other. The contribution of this paper is to develop new instruction model for e-learning, analyze the learning process, map system’s behavior from human growth, and give their perspectives.

The rest of the paper is organized as follows. Section 2 outlines of motivations for this study and investigations in universities. In Section 3, we propose a novel software engineering instruction theory based on analogical thinking. In Section 4, we show the effectiveness of our method by using in actual lectures. After that, in Section 5, we present an example of instruction system that utilizes analogical thinking. Finally in Section 6, we provide some final remarks.