Chapter 5

Tools for the Learning of Programming Languages and Paradigms: Integration of a Code Validator and Exercises Module Into the Moodle eLearning Platform

María A. Pérez-Juárez
University of Valladolid, Spain

Míriam Antón-Rodríguez
University of Valladolid, Spain

María I. Jiménez-Gómez
University of Valladolid, Spain

Francisco J. Díaz-Pernas
University of Valladolid, Spain

Mario Martínez-Zarzuela
University of Valladolid, Spain

David González-Ortega
University of Valladolid, Spain

ABSTRACT

The learning of programming languages and paradigms is complex and requires a lot of training. For this reason, it is very important to detect students’ main problems and needs to be able to provide professors with tools that help students to overcome those problems and difficulties. One type of tool that can be used for this purpose is the code validator. This chapter explores the possibilities and impact of using different tools and strategies for learning programming languages and paradigms. To achieve this goal, the authors have conducted a comprehensive search of relevant scientific literature that has been complemented with their experience using a JavaScript code validator and exercises module integrated into the e-learning platform Moodle, with university students during a web programming course.

DOI: 10.4018/978-1-5225-7455-2.ch005
INTRODUCTION

The learning of programming languages and paradigms is quite difficult and requires a lot of additional work, especially when compared to more theoretical fields of study (Brito & Sá-Soares, 2014). According to Vega et al. (2013) students think that programming is a complicate field of study and it is quite frequent to hear about problems related to frustration and lack of motivation. Moreover, programming is hard to teach; in fact, there are professors that think that programming requires skills other than knowledge (Tan et al., 2014).

Programming students usually have a good knowledge of concepts and theory but experience problems when trying to put those into practice (Newstetter & McCracken, 2001). Research has shown that previous knowledge is very important in a learning process (Chi & Ceci, 1987; Chi et al., 1988; Glaser, 1984). For some students, unfortunately, the foreknowledge they have is wrong and they are reluctant to change those initial ideas. Along this line, Newstetter & McCracken (2001) performed a study for a better comprehension of the nature of computer programming learning. Their study showed that some students have wrong previous ideas, a fact which must be considered by lecturers. In this sense, it is very important to detect students’ main problems regarding programming languages and paradigms learning, in order to be able to provide professors with tools that help students to overcome those problems and difficulties.

Moreover, in order to learn a programming language, it is necessary to be able to write code in that programming language, and not only to answer short questions. This is also the best way to determine the knowledge of students about a specific programming language (Danutama & Liem, 2013).

Besides the inherent difficulty of programming as a discipline, the focus of the problem could be in the use of inefficient and inadequate methodologies and educational tools to teach this subject. For this reason, during the last decades, experts have searched for ways to improve the academic performance of students, especially in the case of newcomer students (Moons & De Backer, 2013).

In such a complex context it is important to have a clear view of which are the main problems that students face when approaching to the theory and practice of programming languages and paradigms. The educational tools oriented to learning programming should consider those needs and provide resources and strategies to manage them.

Taking into account the difficulties related to the teaching and learning of programming languages and paradigms, this chapter explores the possibilities and impact of using different tools and strategies for the learning of programming languages and paradigms. To achieve this goal, the authors have conducted a comprehensive search of relevant scientific literature that has been complemented with their
Teaching Model-Driven Engineering in a Master's Program: Three Editions on a PBL-Based Experience
www.igi-global.com/chapter/teaching-model-driven-engineering-in-a-masters-program/219980?camid=4v1a