Skills based assessment in SystemC modeling

Katarína Jelemenská *, Pavel Čičák

Faculty of Informatics and Information Technologies, Slovak University of Technology, Ilkovičova 3, Bratislava 842 16, Slovakia

Abstract

The paper presents a knowledge and skills assessment approach developed to support Digital system description course in which students should gain the skills in digital systems modeling using hardware description languages (HDL). The previously used course exams in pen and paper form were neither popular nor suitable in this area since they have not allowed the verification of the student’s ability to debug the created model, which is the substantial part of the student’s practical skills. That is why we developed several knowledge assessment tools that enable reliable evaluation of practical skills in HDL modeling without substantial teacher involvement. The tool developed for SystemC language has the form of a new Moodle activity module and enables to edit, compile, and simulate SystemC models. The syntax highlighting editor is integrated directly into the module, however for model compilation and simulation the external tools are invoked by means of shell scripts called from PHP application. The submitted models are automatically assessed based on the simulation results comparison.

1. Introduction

In programming languages courses knowledge and skills assessment has always been an issue no matter what kind of programming language was concerned [1, 2]. The problem is very similar in the case of hardware description languages (HDL). To master the programming or modeling language, the students should solve various problems using the language. It is useless to learn language constructs or exact syntax by heart without writing any program. That is why the lab exercises in programming courses are usually based on more or less individually allocated assignments that the students can solve at school or at home. The solutions are evaluated by teachers and usually form a part of the overall course assessment. Since there is no way to prevent some students from cheating and different teachers might use various scales for assignments evaluation, this part of the course assessment is often neither objective nor realistic, although requiring a lot of teachers’ work. The second part of the course assessment is usually formed by

* Corresponding author. Tel.: +421-910-942-659
E-mail address: jelemenska@fiit.stuba.sk

© 2011 Published by Elsevier Ltd. Selection and/or peer-review under responsibility of [name organizer]
an exam, sometime supplemented by some midterm tests. In order to make the course assessment process more objective, fair and effective, the substantial part of the course assessment should be shifted from the assignments to the midterm and exam tests. However, at the same time, the new techniques have to be used that will allow for the skills based tests assessed in an automated manner [3].

To ease the work needed for knowledge assessment in the course, various Content Management Systems (CMS), Learning Management Systems (LMS), Computer-Based Training (CBT) or Web-Based Training (WBT) systems can be used supporting automated knowledge and/or assignments assessment. Nowadays there are a vast number of such systems. We have analyzed a number of such systems and the capabilities of some of the reviewed ones are summarized in Table 1. The emphasis was placed on the supported question types, test types and automated assessment. As we can see, most of the reviewed systems support several types of test questions but only WebToTest [4] enables students to edit, compile, and debug a program and submit it using a special form. We were not able to find any system supporting skills based assessment in the area of HDL modeling. This fact inspired us to develop several skills assessment systems supporting languages like VHDL, Verilog, or SystemC. The standalone testing application presented in [15] and [16] and the Drag&Drop Moodle module [17] that have been used for student’s knowledge and skills assessment in the last academic year, are two examples of them. The special SystemC testing module, presented in this paper, is one of the latest solutions, suitable for online skills based examination in the area of digital systems modeling using SystemC language. The module is integrated into the Moodle LMS [14] and it is already a part of the set of assessment tools used in the Digital system description course.

2. SystemC testing module design

The testing tool was designed as the Moodle activity module. This implies the client-server architectural model of the system. The LMS Moodle, together with the new module, are installed on the server. This web-based application requires a web server – in this case the Apache HTTP Server is used.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Program question type</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Multi-choice single answer</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Multi-choice multi answer</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>True/False</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Gap filling</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Matching</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Slider</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Arrange objects</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Place object</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Identify correct picture region</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Identify correct object</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Crossword</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>
A database server is also required. Although several types of databases can be used, the MySQL database is probably supported the best. The commercial software ModelSim [18] from Mentor Graphics Company is placed on the server as well. On the client side, a web browser is used to communicate with the server using a specially developed user interface.

2.1. Module functions

The module functions were designed based on the requirements derived from the needs of two types of users – students and teachers. The students are provided with the following functions: (1) SystemC syntax highlighting editor to create the solution, (2) source code syntax verification (compilation), (3) source code simulation, (4) source code submission, and (5) assessment review.

The teachers’ functions include: (1) test assignment creation – textual description, source code core, reference solution and test bench, (2) time limit setup, (3) assignment grading parameters setup, (4) submitted solutions with automated assessment review, (5) manual modification of assessment, (6) solutions export.

The teacher can copy parts of the model solution to the core which will be displayed in student’s source code editing window. Although the core is voluntary, if the teacher doesn’t use it, the system will not usually be able to do automated evaluation. The assignment grading parameters include maximum number of points, difference penalty, and syntactical correctness gain.

The automated tasks are realized by the system. The system must guarantee to store the student’s solution any time it is required to do so. The source code must be stored correctly even though it may contain special characters such as apostrophes, etc. The most important module functions are the source code syntax verification and design simulation. To compile and simulate SystemC code, ModelSim tools are used. The compiler and the simulator can be launched from the command line and therefore initiated from another application. The ModelSim tools are called by the PHP application using console mode and provided source code as an input. ModelSim then processes the provided input and returns the output to the PHP application. The source code is compiled each time a student requires it. The actual compilation is done invoking the `sccom` command that interacts with a C/C++ compiler to compile SystemC source code and links the design afterwards. The compiled model is also simulated each time a student requires it. Simulation is done invoking the `vsim` simulator, an integrated part of ModelSim. The source code submission process is activated by the system either automatically, in case the time allocated to the test elapsed, or interactively by the student’s command. As part of this process the student’s test bench will be replaced by the teacher’s one, and the test will be automatically evaluated, comparing the resulting waveform to the reference. For each difference the setup difference penalty is subtracted from the maximum number of points. In case the source code can be compiled, but there are too many differences, a syntactical correctness gain is returned as a result. In case the source code is not syntactically correct, it can still be compared to the reference solution to find some partial similarities.

2.2. Module user interface

The screenshot shown in Fig. 1 illustrates the student interface to the designed module. Shown at the top of the screen is the test assignment text. Under this part the SystemC source code editing window is situated, where the provided core should be completed by a student. SystemC syntax highlighting, similar to that used by ModelSim (used during the term), is used to make the editor familiar to students. The teacher’s user interface consists of the test creation form. This form is only accessible to the teacher or administrator. The teacher sets all the test attributes here - for example, duration of the test, the testbench, reference solution, etc.
3. Results and experience

The new testing module has been used in the Digital systems description course to evaluate students’ practical skills during the midterm tests. In the course, the students pass three midterm tests and one final exam. On each midterm test, the students are divided into three groups based on the testing environment – TabletPC [16], Drag&Drop [17], and VHDL or SystemC Moodle test. Each student should pass each type of test once. However, the test assignment is the same, regardless of the test type. This constraint ensures that the severity of all the types of the test should be more or less the same. The results of the midterm tests (given in Fig. 2) show that the students reached the best test attainment in the Drag&Drop test. The reason is that in this case the correct solution is already given in the test and the students just have to find the correct statements and place them correctly into the SystemC model. According to the survey results most of the students also think that this test type is the best way of testing practical skills.

4. Conclusion

The first results prove that the kind of online test described previously provides more realistic image of student knowledge of digital systems modeling, compared to the tests based on the commonly used online tests question types. Because the SystemC Moodle test requires creativity by students, the test results are worse compared to the Drag&Drop [17] test. The presented testing solution brings number of advantages. First, there is a substantial reduction in the demanding and time consuming work of the teacher, compared to paper exams correction. At the same time, the level of difficulty of the exam is preserved. Second, it enables the teacher to check the skill of students in the area of model debugging, which was previously not possible. All participants in the examination have at their disposal a source code editor with syntax highlighting, SystemC language compiler, and simulator outputs which makes the model development easier. The application gives the teacher means to comfortably manage the testing system. Finally, the students have the opportunity to verify their designs before sending them for evaluation. The module,
integrated into the Moodle LMS, became an integral part of the assessment tools for Digital system description course.

Acknowledgements

This work was supported by Slovak Science Grant Agency - VEGA 1/1008/12.

References