PHPi – An environment to support web programming learning using PHP

Ricardo Antunes[1], and António José Mendes[2]

***Abstract* - The increasing Internet popularity gives an extra importance to web programming. Many universities world wide have included this topic in their Computer Science curricula. It usually appears after students pass introductory programming courses. Although basic programming knowledge is essential, web programming adds some specific aspects that students must address. This topic embraces so many different technologies that students often reveal learning difficulties. To support our students' learning we developed an environment, PHPi – PHP interactive, designed to support educational activities of web programming learning using the PHP language. PHPi is essentially a visualization tool that allows students to create PHP programs and observe the execution of server code. They can relate program instructions with their results, allowing a better program understanding. In this paper we describe PHPi in more detail and give a utilization example.**

Index Terms – Web programming learning, PHP language, learning environments.

INTRODUCTION AND MOTIVATION

Since its appearance Web is continuously changing the way people communicate, spread information, do business, etc. The early days of static HTML pages belongs to the past. Currently web information needs to be dynamically updated and delivered with a minimum effort. This reality stimulates the development of dynamic web sites and powerful web development tools and programming languages which combined with the ever increasing popularity and pervasiveness of the web makes this subject an important topic for an academic course as referred by [6, 10].

Despite the high motivation and interest students may have, learning how to program can be a very difficult and complex task to achieve [1, 3, 4, 5, 7, 8]. Usually web programming is not taught in introductory programming courses but it adds specific concepts that are not addressed in those courses, such as the coexistence of several programming languages within the same file, client-server programming, and database access.

Many authors have the opinion that pedagogical activities must actively engage students to achieve better learning results. In particular learning how to program is a subject where students must acquire several competencies far beyond the mere memorization of the language syntax. It is generally accepted that practise is vital to learning programming. For instance [5] state that "it is important for the learning that the students do programming by themselves" revealing the high importance an active pedagogical activity may have where students actively construct their programs in opposition to passively viewing already made solutions.

Some studies were done about the effectiveness of algorithm visualization in learning like the one performed by Stasko, Badre, and Lewis [9]. This study showed somewhat discouraging results. Instead of what was expected the results revealed that the difference between students that used both textual materials and algorithm animations and students that used only text materials was not statistically significant. One reason cited by the authors for these results could be the passive mode the viewer engaged students. Authors conjecture that an active student engagement where students construct their own animations may benefit learning. Animations should also be accompanied by teacher explanations about what's being viewed.

More recent studies such as [2] reinforce the idea that active student engagement benefits learning.

In this paper we present PHPi, a web based environment designed to help our students to overcome the difficulties of learning web programming concepts. PHPi is a visualization tool which allows students to watch and analyze the execution results of a given PHP program. Programs are executed on the web server and results are shown on the students' web browser. It is also possible to type in code enabling students to create, modify and execute their own solutions so they can reflect about the execution results and make the necessary code corrections, if needed.

PHPI DESIGN GOALS

The following goals guided the development of PHPi:

- PHPi is intended to support server-side web programming learning focusing server to client program output.
- It is assumed that students already have introductory programming instruction.
- Students may use the environment outside classes.
- The environment should present different types of exercises to better support different learning stages.
- Students should be actively engaged to achieve better learning results.

[1] Ricardo Antunes, Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria - Portugal, antunes@estg.ipleiria.pt
[2] António José Mendes, Centro de Informática e Sistemas da Universidade de Coimbra, toze@dei.uc.pt

Two important issues to have in mind are student usage rejection and the time it requires students to install and begin using the environment. We think PHPi's features incentive its use namely:

- **Simplicity**: the environment is based on simple web pages similar to other student's familiar web applications. Information is organized and shown as needed so that students don't feel lost and confused.
- **Interactivity**: students have an active learning role interacting with the environment. They can create, test and correct their own problem solutions in opposition to passive viewing of materials.
- **Cost**: PHPi does not require the acquisition of additional hardware than the one already being used. As it was developed using free software it also doesn't require extra software licenses.
- **Availability**: Students can use it inside as well as outside classroom according to each student learning requirement time.
- **Installation and update**: It only requires students to have a web browser and a web connection. Additional software installation is not needed. Future updates are performed on the web server eliminating the need to distribute individual updates.

THE PHPI ENVIRONMENT

PHPi is a client-server web application designed to support server-side web programming learning concepts using PHP language. When a student starts the environment she/he is presented a menu with the available topics. For each topic there are pages of information and exercises. PHPi gives special importance to student practise through exercises. There are three types of exercises available, each requiring different levels of knowledge and pedagogical activity.

If a student is not ready to solve an exercise because she/he couldn't be present in class or requires extra time to study some topic due to student knowledge background variations, she/he can obtain that required information from the available information pages. It can be also useful to reinforce acquired information or clarify any doubt students may have. Thus the environment does not restrict its usage to class time minimizing traditional instruction time limitations.

I. Types of exercises

PHPi supports three types of exercises: annotated example programs, program sorting, and coding exercises.

1. **Annotated example programs** are useful when students aren't ready to code their own programs yet. Teachers can create annotated example programs that students can follow step by step. On each step students are presented with server code, execution output, and additional information, all provided by the teacher. The information detail may vary on each step and is shown to the student when that particular step is executed focusing students attention on particular aspects considered relevant by the teacher. On each step students are able to move forward or backward, so that they can review any step they didn't correctly understood.
2. **Program sorting** is an intermediate type of exercise that can be used when students are able to understand code but aren't still able to create their own programs. Teacher gives an unsorted program and asks students to sort the instructions correctly according to a given purpose. Students must understand each one of the code lines and how they are related. After sorting the code students may submit their solution for correctness evaluation. If the submitted solution is wrong students receive appropriate feedback and they can correct their solution accordingly. If the submitted solution is correct it is executed on the web server and the output is shown to the student.
3. **Coding exercises** allow students to construct their own problem solutions to proposed exercises. Using the environment students can develop, test, correct and improve their own programs. They can see each instruction results on the program's variables and output, allowing them to verify if the program works as they expected. After students submit a program the environment makes some tests and constructs the execution output. If the program doesn't pass the tests students receive adequate feedback, so that they can locate errors easily and make the necessary corrections. Students can follow the program execution outputs step by step. We believe that program creation and debugging (and not only program reading and understanding) are very important for the development of students' programming competences.

II. Tests performed on student's code

When the student submits a coding exercise it must pass some tests before the construction of the program execution animation takes place.

The first test performed by the environment is function usage check. The student must only use functions that are available for each exercise. Those functions are chosen by the teacher when creating each exercise and are shown to the student. The purpose of choosing a set of functions is to give students a starting point and to focus them on what is considered important to that particular exercise. Other important issue is to avoid students to use some functions that have nothing to do with the exercise and could be dangerous to the environment as trying to read/write environment files. The decision of what functions should be available is up to the teacher.

When solving an exercise students may use a subset of the available functions unless the exercise can only be solved using all of them. We think when conceiving an exercise the teacher should have in mind more than one possible exercise solution and provide the set of available functions accordingly so that the exercise solution doesn't be limitative.

For each exercise the teacher can also indicate some mandatory functions to be used, for instance if the exercise is about a database query a database connection must be

performed before the query. So the second test is to verify if students make use of mandatory functions.

The third test performed by the environment is syntax correctness. This test consists on executing the program and verifying if there is any resulting syntax or run-time errors.

UTILIZATION EXAMPLE

In this section we present a coding exercise utilization example. Imagine that a student wants to learn about how to perform a mySQL database query. She/He can select an exercise form the menu. After doing that the student is presented a window with four different areas: messages, PHP code, available functions, and controls.

The purpose of the exercise is shown on the *messages* area. Code is typed in by the student using the *PHP code* area. Figure 1 shows an example. Read only and editable code may be provided by the teacher and students are required to fill in the gaps. In this case the database parameters and the query string are provided as read only code. To solve the proposed problem students can only use functions displayed on the *available functions* area. Each function name displayed on that area is a hyperlink to an existing explanation of that function so that students may clear any doubt that may arise.

At any time students can obtain help about how to solve the exercise using the *help* button.

When a student wants to test the program she/he presses the *submit* button on the *controls* area. The test takes place on the web server. Students receive feedback accordingly.

In this example a database server connection is not established. A mandatory function (mysql_connect) is missing and so the environment alerts the student about that missing function as seen in figure 2. After correcting the problem the program can be submitted again. Another problem exists when executing the program - the myqsl_fetch_row parameter is not specified. This error is also shown to the student so she/he can correct it. Figure 3 illustrates what happens when the code is submitted. The response window bar is red in order to better distinguish it from a successful situation.

After the submitted code passes all the tests it is executed and the student is presented with the program execution visualization as shown in figure 4. Students can see the server to client HTML response and the step by step program execution animation. The response window bar is now blue showing that there is no syntax/execution error situation. In this utilization example the response refers to a HTML page with a 2x2 table with some data on it.

Figure 5 shows the animation window. In this window the student can see the submitted code, the output generated by each line and the program variables' names and values.

Students can move forward or rewind to the visualization previous state using the *previous* and *next* buttons. On each visualization step the line of code and the corresponding output are highlighted allowing a better understanding of its relation.

The *close* button closes the animation and returns to the exercise window.

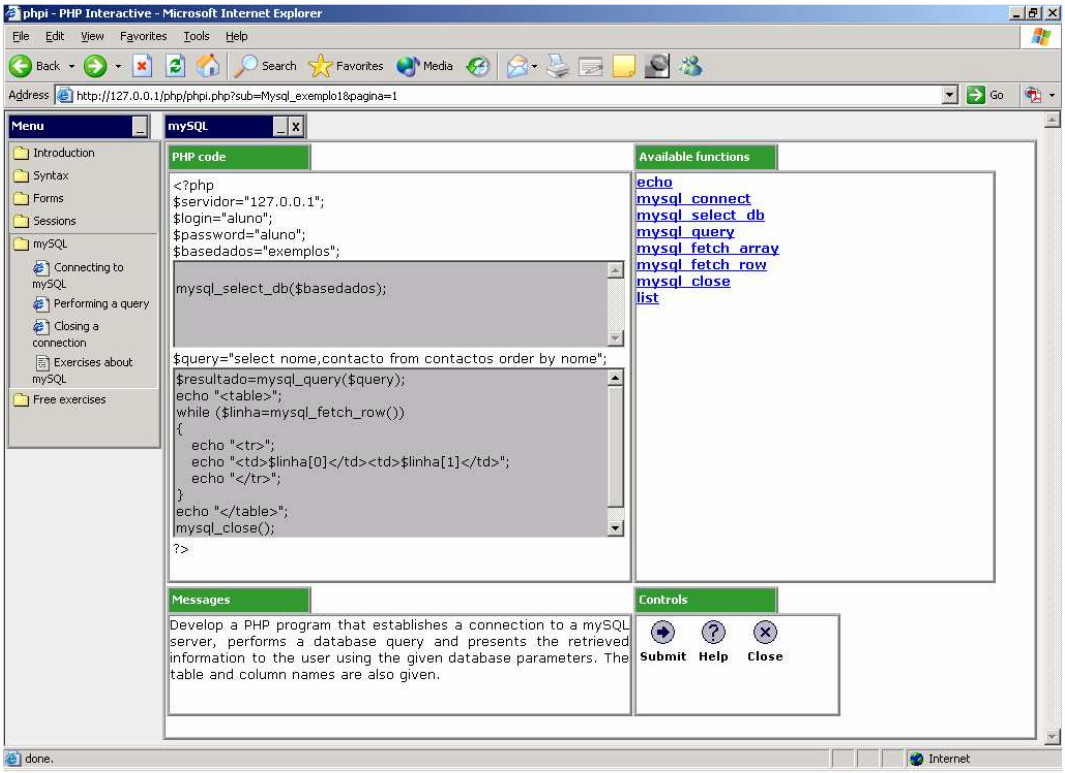

FIGURE 1
CODING EXERCISE EXAMPLE

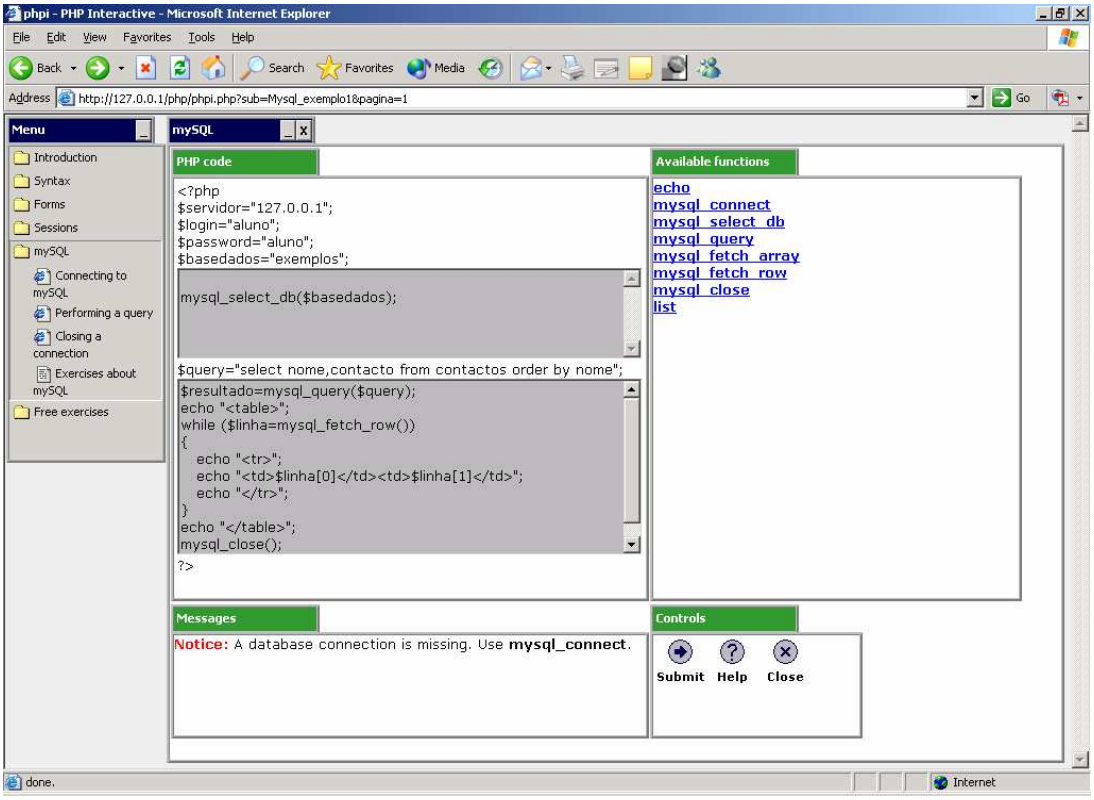

FIGURE 2
MISSING MANDATORY FUNCTION MESSAGE EXAMPLE

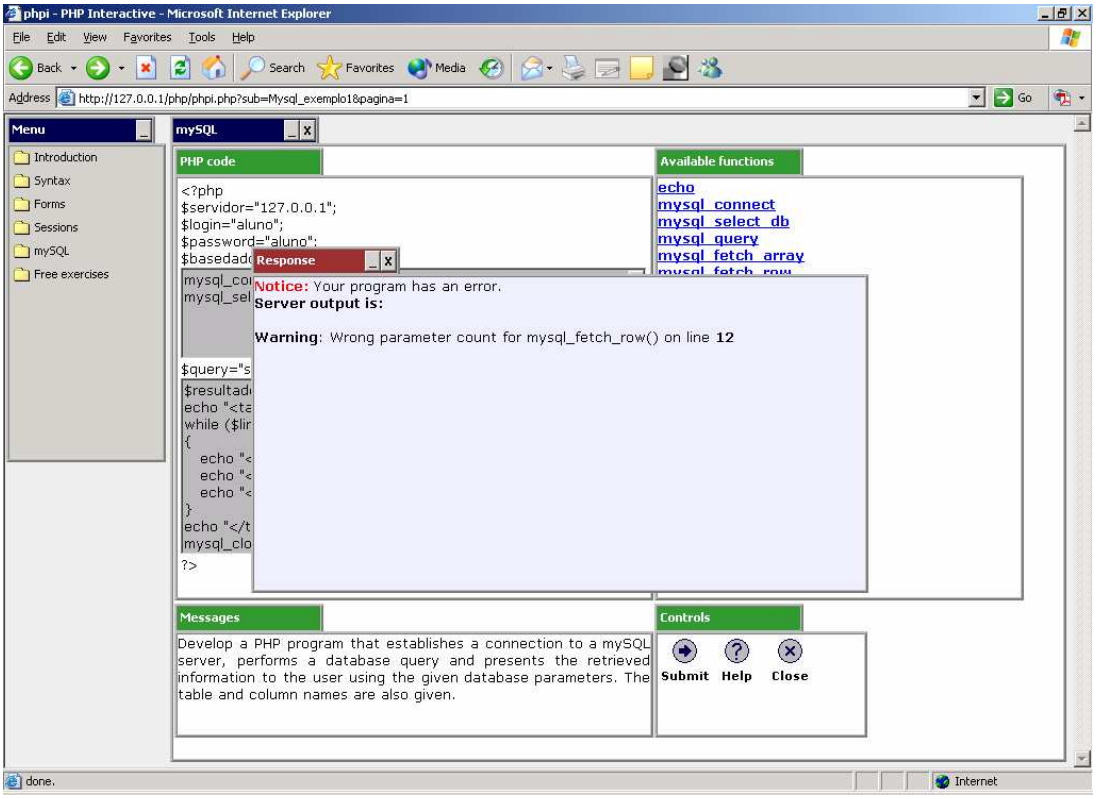

FIGURE 3
EXECUTION ERROR MESSAGE EXAMPLE

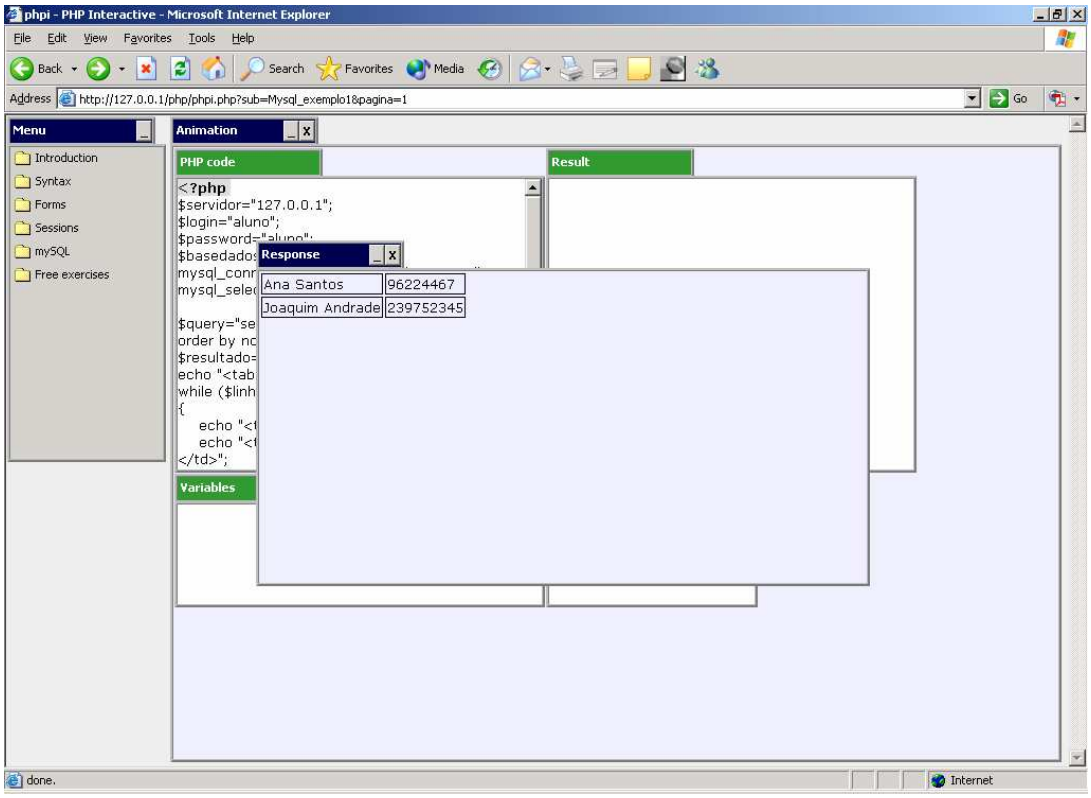

FIGURE 4
SERVER TO CLIENT HTML RESPONSE EXAMPLE

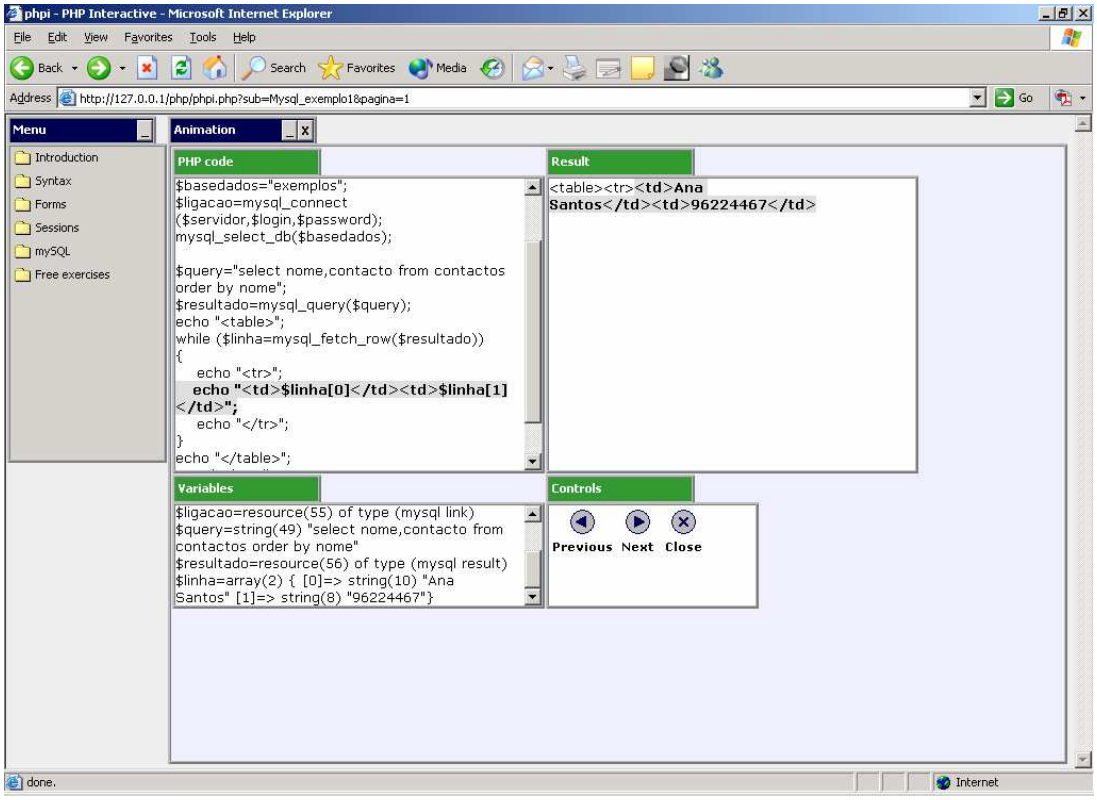

FIGURE 5
PROGRAM EXECUTION ANIMATION EXAMPLE

CONCLUSION

Learning to program is a difficult and challenging task that requires time and dedication. It is very important that students do programming by themselves. The use of computers as a learning aiding tool can be very helpful specially if teachers create and carefully select pedagogical approaches and materials.

The environment presented in this paper tries to minimize students' difficulties and traditional instruction limitations and provides learning activities directed to student practise. Students are able to use it during class time as well as outside classroom.

Some preliminary tests have been made mostly with teachers responsible for web programming courses. However we are aware that student evaluation is essential to take conclusions about the environment educational effectiveness and about problems and/or improvements that can be made. We plan to conduct that evaluation in the first semester of the next academic year during our web programming course.

REFERENCES

[1] Bonar, J., and Soloway, E., "Uncovering principles of novice programming", Proceedings of the 10th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, 1983, pp.10-13

[2] Hundhausen, C., Douglas, S., and Stasko, J., "A meta-study of algorithm visualization effectiveness", Journal of visual languages and computing, 13, 2002, pp. 259-290

[3] Kelleher, C., and Pausch, R., "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers", ACM Computing Surveys (CSUR), 37(2), 2005, pp 83-137

[4] Kujansuu, E., and Kulmala, M., "Codewitz: producing interactive elearning material for beginners in programming", Proceedings of the 8th annual Conference on Innovation and Technology in Computer Science Education ITiCSE'03, 2003, pp. 266-266

[5] Lahtinen, E., Ala-Mutka, K., and Järvinen, H., "Early Programming: A Study of the Difficulties of Novice Programmers", *Proceedings of the 10th annual SIGCSE Conference on Innovation and Technology in Computer Science Education ITiCSE'05*, 2005, pp. 14-18

[6] Lee, A., H., "A Manageable Web Software Architecture: Searching for Simplicity", SIGCSE'03, 2003, pp. 19-23

[7] Mendes, A., Gomes, A., Esteves, M., Marcelino, M., Bravo, C., and Redondo, M., "CS1-2: Using Simulation and Collaboration in CS1 and CS2", Proceedings of the 10th annual SIGCSE Conference on Innovation and Technology in Computer Science Education ITiCSE'05, 2005, pp.193-197

[8] Sanders, I., and Gopal, H., "AAPT: Algorithm animator and programming toolbox", ACM SIGCSE Bulletin, 23(4), 1991, pp. 41-47

[9] Stasko, J., Badre, A., and Lewis, C., "Do algorithm animations assist learning? An empirical study and analysis", Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 1991, pp. 61-66

[10] Treu, K., "To Teach the Unteachable Class: An experimental course in Web-based application design", Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education SIGCSE'02, 2000, pp. 201-205