iProg: Development of Immersive Systems for the Learning of Programming

Manuel Ortega
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Manuel.Ortega@uclm.es

Miguel A. Redondo
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Miguel.Redondo@uclm.es

Ana I. Molina
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
AnaIsabel.Molina@uclm.es

Crescencio Bravo
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Crescencio.Bravo@uclm.es

Carmen Lacave
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Carmen.Lacave@uclm.es

Yoel Arroyo
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Yoel.Arroyo@uclm.es

Santiago Sánchez
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Santiago.Sanchez@uclm.es

M. Ángeles García
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
MariaAngeles.GMarin@uclm.es

César A. Collazos
Grupo IDIS
Universidad del Cauca (Colombia)
ccollazo@unicauca.edu.co

Javier Jiménez Toledo
Facultad de Ingeniería
Institución Universitaria CESMAG (Colombia)
jajimenez@iucesmag.edu.co.

Huizilopoztli Luna-García
Unidad Académica de Ingeniería Eléctrica
Universidad Autónoma de Zacatecas (México)
hlugar@uaz.edu.mx

J. Ángel Velázquez-Iturbide
Dpto. de Informática y Estadística
Universidad Rey Juan Carlos (Spain)
angel.velazquez@urjc.es

Raúl Abad Gómez-Pastrana
Escuela Superior de Informática
Universidad de Castilla – La Mancha (Spain)
Raul.AGomezPastrana@uclm.es

Interacción '17, September 25–27, 2017, Cancun, Mexico

ACM ISBN 978-1-4503-5229-1/17/09…$15.00
https://doi.org/10.1145/3123818.3123874

ABSTRACT

Basic "computing literacy" is said to be deemed necessary for all citizens, and provides an opportunity to prepare, over longer periods of time, future computing engineers. The iProg Project intends to achieve computing literacy research objectives by means of a number of applications for *Programming Education*, based on different techniques for advanced Computer-Human Interaction and visualization, including

augmented reality and gesture-based interaction. The evaluation of Usability and User Experience of these new forms of interaction will require the use of advanced interaction techniques, including eye tracking and the gathering of biometric information.

Author Keywords
Research in learning programming, eLearning, interactive systems, human-computer interaction, evaluation.

ACM Classification Keywords
H.5.m. Information interfaces and presentation. D.1 Programming techniques.

D.2.6 Programming environments. Interactive environments.

INTRODUCTION

In the last decades, learning mechanisms have progressed, which in synthesis can be grouped in three main advances:

1. Revolution of Informatics and Communications. ICT (Information and Communication Technologies) continuously changing our society, including industrial and business changes to social relations. This revolution also affects education. The maturity of web technologies has generalized online education through e-learning platforms. Other important innovations are video games, MOOCs, CSCM (Collaborative Supported Collaborative MOOCs) [1] as well as new mobile (phones, tablets, etc.) and wearable devices.
2. The Pedagogy Process. Since John Dewey began an education based on experience, several pedagogical theories have emerged. Theories centered on knowledge transmission (behaviorism) have given way to others based on their construction by students (constructivism and constructionism). Associated with pedagogical theories, different forms of *active learning* have emerged, such as collaborative learning.
3. Promoting the teaching of STEM (Science, Technology, Engineering and Mathematics) at pre-university levels. Recently, sub-tending has claimed the extension of Informatics teaching to pre-university levels.

Therefore, the trends and developments mentioned above cannot be ignored, nevertheless, they must not be assumed as insignificant. In this sense, we will mention some concerns:

1. Educational Informatics takes on new forms that sometimes presume a regression in the underlying pedagogical theories. For example, the first MOOCs provided passive instructions, similar to classroom instruction.
2. The benefits of educational tools developed are not always evaluated, based on the latest technological developments, nor of the alternative didactic proposals, without experimental evidence of their well being, without being integrated into teaching practices.

Of course, these criticisms are not general in nature and their relevance in each case must be analyzed. A major difference between Computer Science learning and that of other disciplines (such as Physics or Mathematics) is that they have a long didactic tradition. In the case of Computing, there are numerous individual contributions (agendas, tools, etc.), but without the proper rigorous Pedagogy research evaluation to research in Pedagogy and without consolidated didactic theories cannot be evaluated.

However, in the last 15 years there has been a strong trend that proposes a greater research rigor in the area of teaching Computing [2]. This effort has been concentrated mainly on Programming learning, for its introduction and central role in Computing. This research trend is also seen in other engineering fields, represented by ASEE (American Society for Engineering Education) and IEEE (Institute of Electrical and Electronic Engineers). However, it has been ACM's SIGCSE (Special Interest Group in Computer Education) which has given the greatest impetus to computing teaching as a research field (sometimes called Computing Education Research, CER).

In this context, it is necessary to pay attention to the new Computing teaching phenomenon at pre-

university levels, given the undeniable social interest and vigor of this trend.

Let us briefly review some related aspects in order to better appreciate the study state and possible research opportunities.

APPROACHES AND TOOLS FOR PROGRAMMING LEARNING

Programming is a difficult discipline to learn. Computer scientists, pedagogues and psychologists have spent several decades researching the difficulties that students have in learning [3]. Studies have been focused mainly on procedural and objects oriented to programming, sometimes with an emphasis on the most difficult concepts, such as recursion and inheritance. We have identified the ways in which students understand (in a non-viable way) the concepts [4], Also didactic methods have been proposed [3] and educational software tools have been developed [5].

Our point of view is not only constructivist but also constructionist, that is, we assume that students learn by constructing *artifacts*, in this case programs or algorithms. A huge variety and number of systems have been developed for the learning of the Programming at university levels. In a very simplified way, we could distinguish two kinds of systems [5]: (1) Most of the systems used to learn programming in order to help understand Programming *mechanics*, either static or dynamic; (2) Other systems let students learn to program while performing some tasks that interests them, for example, gaming.

Taking as reference the first type of systems (used at university levels or in specific training cycles), it can be seen in learning Algorithm overlaps, and is sometimes confused with that of Programming. However, it is assumed that university students who learn Algorithms have basic Programming dominance. Therefore, the algorithm is usually a more advanced course, which requires more advanced tools. It is typical to use visualization systems and measure their performance. In this sense, augmented reality and gesture-based interaction materialize immersive interaction styles [6] which, together, can enhance the benefits of constructivist and situated learning theories. [7]. Fundamentally, these styles of interaction bring students closer to the reality of the domain under study and, among others, promote contexts of active learning and for discovery.

At pre-university levels and for learning Programming, the second class of systems identified is of greater importance [8], in the form of micro worlds or games, as well as the use of scaffolding (e.g. through conversational agents [9]). Not having a clear definition of educational objectives as in the university, it is customary to adopt other approaches. For example, the combination of hardware and software is common with basic forms of robotics or electronic controllers. It is also worth noting the increasing use of forms of interaction, different from the usual ones based on graphical desktop interfaces. We usually talk about "natural" user interfaces [6] based on touch, vision or speech, such as gestural devices (e.g., Kinect), multi-contact tables, immersive systems and wearable devices.

In the scope of our research group (CHICO, Computer Human Interaction and Collaboration) we have made some developments in the field of teaching/learning Programming. Our current objective is to adapt its use to new educational levels, as well as to exploit the use of new paradigms of interaction in such environments. Some of the newly developed systems are described below:

The first project is COLLECE system [10], this one was created to explore the benefits of the Collaborative Programming, in teaching-learning scenarios as professionals. COLLECE allows groups of remote users to edit programs simultaneously and to resolve previously defined programming tasks. To do this, the system provides shared workspaces for editing, compiling, and running programs with collaboration support tools. The system has been used in different experiments [10,11] with programming problems of moderate complexity,

such as ordering algorithms and recursion, and of greater complexity, such as distributed programming and systems with multi-level architecture.

The **VisBack** application [12] (Figure 1), allows graphical visualization *of* the trace of the execution of *backtracking* algorithms based on combinatorial techniques. For this, the graphical representation of the trees generated by the different recursive calls is used, showing in each node the information that the user chooses. Thus, a clear and simple visualization is obtained for the students. It is a visualization system with

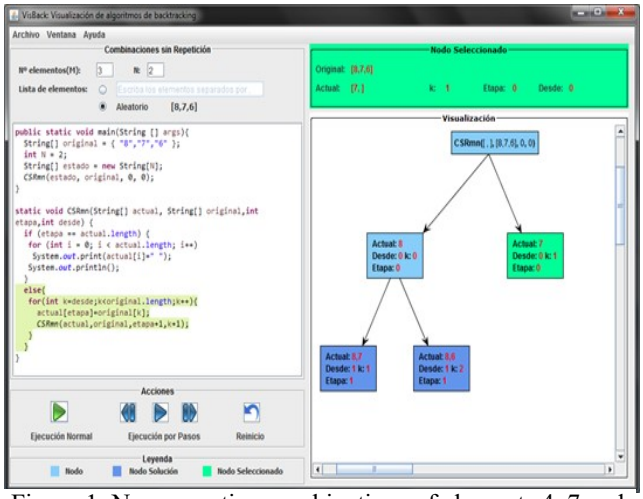

Figure 1. Non-repeating combinations of elements 4, 7 and 6, taken 2-in-2 in VisBack with advanced node visualization

certain elements in common with **SRec** [13], but focused on backtracking algorithms, instead of visualizing the recursion in a generic way.

TOWARDS THE LEARNING OF PROGRAMMING WITH IMMERSIVE TECHNIQUES: THE ENVIRONMENT COLLECTS 2.0.

One of the developments in which the CHICO group is currently working on is the **COLLECE 2.0** environment, based on the earlier project, **COLLECE**.

COLLECE 2.0 is an environment based on the Eclipse IDE for which an extensive set of plugins has been developed for Programming [14,15,16], which allows collaborative editing of code in JAVA and other Programming languages. Figure 2 shows the collaborative environment, in which three participants edit code and chat in an integrated environment and elaborated ad hoc for this purpose in the project. The use of telepointers and other awareness mechanisms, allow programmers to have information at their fingertips at all times, depending on what participants do in the collaborative code editing session.

Presently, we are working on incorporating new features, which are considered innovative. We refer to plugins that allow (1) 3D algorithm/code visualization of static representations which are being edited; (2) 3D algorithm/code dynamic behavior is being worked on, and (3) interaction through augmented reality techniques with structural representation of the algorithm and its behavior.

Surely, the organization in plugins facilitates the different visualization and interaction elements mentioned above can be included and activated, according to the type of learning task to be carried out in each case.

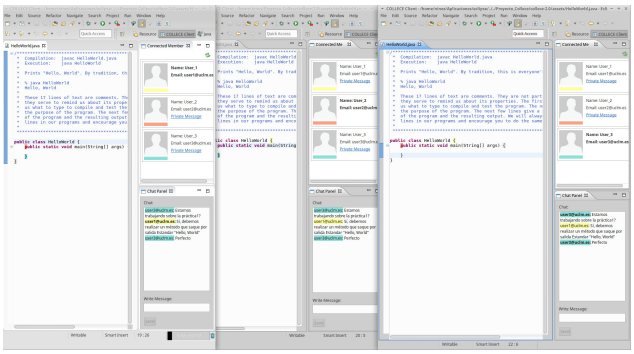

Figure 2. Collaborative editing and chat in the COLLECE 2.0 environment

Additionally, developing tools are within the project which are used in the development of mobile educational applications based on CIAM development methodology [17,18] (Figure 3). These applications will be used within the same project in the near future. Finally, these developments will be evaluated with *eye tracking* techniques, according to the experience of the CHICO group in these techniques [19], which will result in the improvement of tools themselves, the validation of evaluation techniques and the evolution of the evaluation

methodology and analysis of the results obtained in the learning activities that can be proposed by these types of tools.

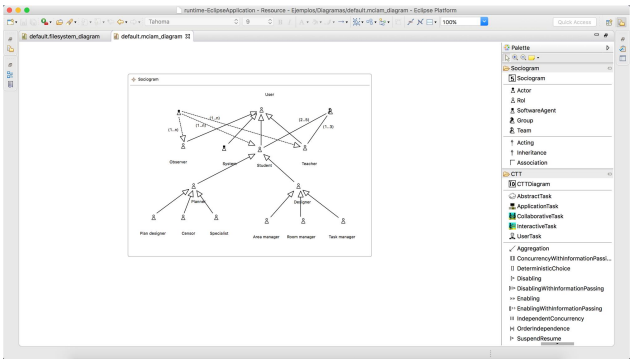

Figure 3. Diagram editing (sociogram) with CIAM methodology support tool

CONCLUSIONS

The use of usability analysis objective techniques such as eye tracking as well as model-based development techniques and the CIAM development methodology have allowed the CHICO group to develop a set of educational and collaborative applications aimed mainly at learning programming within a university environment.

Currently we propose to add new tools and functions that allow students to improve their interaction with the learning systems through immersivity using 3D representation and Augmented Reality techniques.

AGRADECIMIENTOS

The Ministry of Economy, Industry and Competitiveness has partially financed the present study through Project TIN2015-66731-C2-2-R.

REFERENCES

[1] Collazos, C., González, C., García, R., Computer Supported Collaborative MOOCs: CSCM, Proceedings of the 2014 Workshop on Interaction Design in Educational Environments, 2014.

[2] Fincher, S. & M. Petre (eds.) (2004). *Computer Science Education Research*, Routledge

[3] Robins, A., Rountree, J. & Rountree, N. (2003). "Learning and teaching programming: A review and discussion", *Computer Science Education*, **13**(2):137-172

[4] Clancy, M. (2004). "Misconceptions and attitudes that interfere with learning to program". En M. Petre & S. Fincher (eds.), *Computer Science Education Research*, Routledge, cap. 1

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

[6] Bolter, J., Engberg, M. & MacIntyre, B. (2013). "Media studies, mobile augmented reality, and interaction design", *Interactions*, **20**(1): 36-45

[7] Dunleavy, M. & Dede, C. (2014). "Augmented reality teaching and learning". En *Handbook of Research on Educational Communications and Technology*, J.M. Spector *et al.*, (eds), Springer New York, pp. 735-745

[8] Riojas, M., Lysecky, S. & Rozenblit, J. (2012). "Educational technologies for precollege engineering education", *IEEE Trans. Learning Technologies*, **5**(1):20-37

[9] Johnson, W., Rickel, J. & Lester, J. (2000). "Animated pedagogical agents: Face-to-face interaction in interactive learning environments", *Journal of Artificial Intelligence in Education*, **11**: 47-78

[10] Bravo, C., Duque, R. & Gallardo, J. (2013). "A groupware system to support collaborative programming: Design and experiences", *Journal of Systems and Software*, **7**(86): 1759-1771

[11] Molina, A.I. *et al.* (2014). "Evaluating the awareness support of COLLECE, a collaborative programming tool", *Proc. XV*

International Conf. Human Computer Interaction, C. González *et al.* (eds.), pp. 1-2

[12] Lacave, C.; Molina, A.I.; Cardona, S.A.; Pérez, J.F. (2015) "VisBack: Un prototipo de herramienta colaborativa para el aprendizaje de la recursividad", Revista do Departamento de Inovaçao, Ciência e Tecnologia, 6:21-24.

[13] Velázquez-Iturbide, J.Á; Pérez-Carrasco, A. (2016). "Systematic development of dynamic programming algorithms assisted by interactive visualization", *Proc. 21st Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2016)*, 2016, pp. 59-64.

[14] Jurado, F. *et al.* (2013). "Cole-Programming: Shaping collaborative learning support in Eclipse*"*, *IEEE-RITA*, **8**(4):153-162

[15] Jurado, F. *et al.* (2009). "Learning to program with COALA, a distributed computer assisted environment", *Journal of Universal Computer Science*, **15**(7): 1.472-1.485

[16] Jurado, F., Redondo, M.A & Ortega, M. (2012). "Blackboard architecture to integrate components and agents in heterogeneous distributed eLearning systems: An application for learning to program", *Journal of Systems and Software*, **85**(7):1.621-1.636

[17] Molina, A. I., Redondo, M. A., Ortega, M., & Hoppe, U. (2008). CIAM: A Methodology for the Development of Groupware User Interfaces. *Journal of Universal Computer Science, 14*(9), 1434-1446.

[18] Molina, A. I., Redondo, M. A., & Ortega, M. (2014). Model-Driven Development of Interactive Groupware Systems: Integration into the Software Development Process. *Science of Computer Programming, 89, Part C*, 320-349. doi:10.1016/j.scico.2014.02.030

[19] Molina, A. I., Redondo, M. A., Ortega, M., & Lacave, C. (2014). Evaluating a graphical notation for modeling collaborative learning activities: A family of experiments. *Science of Computer Programming, 88*, 54-81. doi:10.1016/j.scico.2014.02.019