


## Review

# Computer-Human Interaction and Collaboration: Challenges and Prospects

Manuel Ortega 

Escuela Superior de Informática, University of Castilla-La Mancha, 13001 Ciudad Real, Spain;  
Manuel.Ortega@uclm.es; Tel.: +34-926-295481

**Abstract:** Through a series of projects carried out by the Computer–Human Interaction and Collaboration (CHICO) group of the University of Castilla-La Mancha, some proposals are presented to improve the current e-Learning systems by making use of different paradigms of human-computer interaction. Synchronous and asynchronous collaborative systems, ubiquitous computing, and augmented reality can improve the current learning environments. The use of artificial intelligence mechanisms for both learner support and assessment complements these techniques. Emphasis is also placed on the use of automatic application generation techniques using models.

**Keywords:** human-computer interaction; CSCL; CSCW; ubiquitous computing; e-Learning



**Citation:** Ortega, M.  
Computer-Human Interaction and  
Collaboration: Challenges and  
Prospects. *Electronics* **2021**, *10*, 616.  
<https://doi.org/10.3390/electronics10050616>

Academic Editor: Jun Dong Cho

Received: 29 January 2021

Accepted: 3 March 2021

Published: 6 March 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

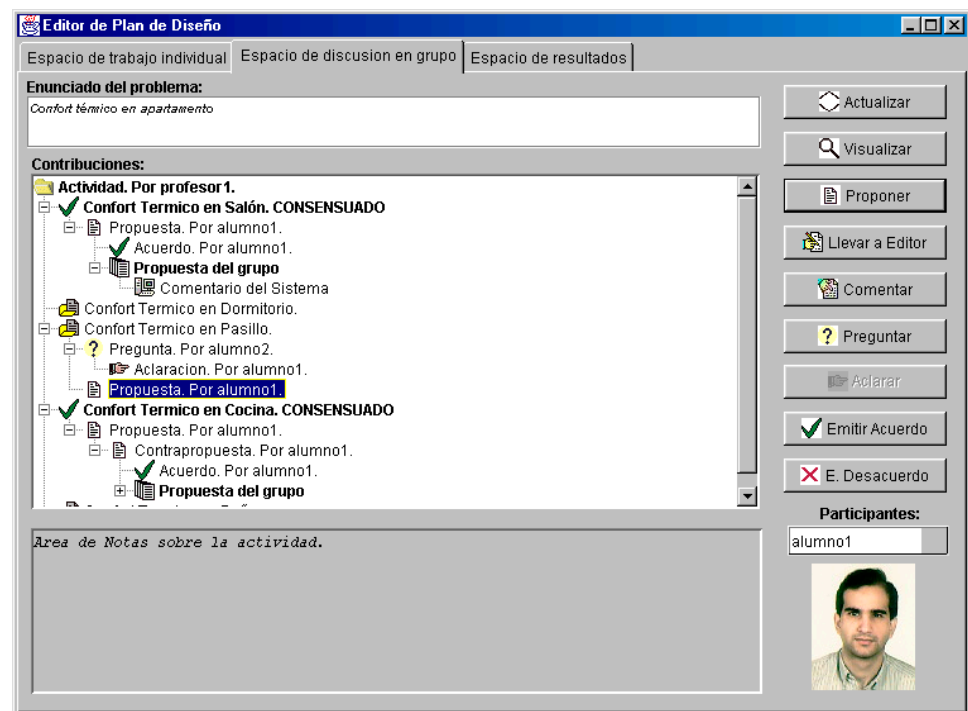
Computers have been used for decades to teach and learn in many different branches of knowledge. Since the year 2020 and as a consequence of the COVID-19 pandemic, their use has become widespread, as they are, in many cases, the only way to attend classes, giving rise to the generalized use of asynchronous and synchronous collaborative tools that have been significantly improved since the 1990s. In an article in 2000 [1], Manuel Ortega predicted the predominance in the immediate future of collaborative systems and ubiquitous computing in eLearning from the perspective of three factors that would participate in the process: teachers, software, and hardware. In the article, the author comments on the necessary changes in the software, taking into account the following [1] (pp. 14): "On the one hand, in order to foster knowledge acquisition in a constructivist way, software should be developed aiming at real and complex projects with problem "scaffolding" and the use of the so-called "Artificial Intelligence" techniques in order to guide the students without overwhelming them. On the other hand, we must focus the solution of those problems in a collaborative way." Of the three factors mentioned above, the teacher component highlighted by Ortega has proved to be of vital importance in this crisis, and his dedication and effort in the transition from face-to-face to online classes have allowed teaching work to continue during the ongoing pandemic [2].

This article aims to review the advances of the last 20 years of research in human-computer interaction, in the areas of Collaborative Systems and ubiquitous computing in e-Learning systems, through research carried out within the Computer–Human Interaction and Collaboration [3] (CHICO) group of the University of Castilla-La Mancha in Spain, which we believe can show the evolution of these systems over the years.

The rest of the article is structured as follows: Chapter 2 presents the basic lines of Computer Supported Collaborative Learning (CSCL) systems; Chapter 3 explains the paradigm of ubiquitous computing; Chapter 4 relates to the use of engineering methods for the creation of user interfaces; Chapter 5 describes the usability of the applications; finally, some developments within the CHICO group related to the teaching of programming are presented, ending with some conclusions and proposals for the development of efficient teaching-learning systems that take into account what has been learned with the discipline of human–computer interaction.

## 2. Computer Supported Collaborative Learning

Collaborative learning systems are divided, in one of their most commonly used divisions, into asynchronous and synchronous. Asynchronous tools are used at different times by the participants in the interaction. A typical example of this kind of system is email. In this type of application, users utilize the tool at different times to send messages to each other, and it is not necessary to reply immediately to the received email. When the email account is opened, they can answer the received emails immediately. By not having to reply in real time, the user can think about how to respond to the asynchronous proposed action. Asynchronous tools, therefore, are used in many cases to plan solutions to complex problems in e-Learning systems, in many cases requiring artificial intelligence mechanisms. As an example, we can propose the PlanEdit tool [4] (see Figure 1), which was developed by students to solve domotics problems within the DomoSim-TPC (Domotics Simulation—Telematic Planned Collaborative) system [5]. In this case, the student solves a planning problem where they can be tutored by artificial intelligence mechanisms. In this asynchronous system, the students select actions individually in the “Individual Workspace”, which are then shared in a “Collaborative Workspace” where they discuss possible solutions. When they reach a consensus, and if the solution testing mechanisms used by artificial intelligence are in accordance with what was planned, the proposal is moved to the so-called “Results Space”. If the discussion in the collaborative space is too long, the system can suggest changes according to the level of scaffolding that the problem allows. The scaffolding helps the student more in the initial problems, even if they are easier, and less in the later problems presented, even if they are more difficult. In this way, help is offered according to the level of knowledge acquired by the student.



**Figure 1.** Asynchronous Computer Supported Collaborative Learning (CSCL) environment of DomoSim-TPC.

Synchronous environments [6,7] (see Figure 2) are those in which students collaborate to solve a problem at the same time, and usually from different locations. These systems usually use different forms of awareness [8] to maintain the feeling of working collaboratively, so each participant knows at all times what the other participants are doing. Figure 2 shows two students collaborating to solve a home automation problem. One of the awareness methods is the use of telepointers—red in the case of the student alumno2

and green for alumno1 (see Figure 2). In synchronous systems, the problems are solved in a limited time and therefore with decisions that are made immediately, thus requiring the solution to be planned in many cases. In this way, synchronous environments should be used after a stage of asynchronous collaboration, which allows for greater reflection on the solution of the proposed problem. These systems consider the individual contributions of the students and the collaboration between them, as well as whether the collaboration improves the results for the proposed problems, using a large number of metrics that reveal the students' levels of development.

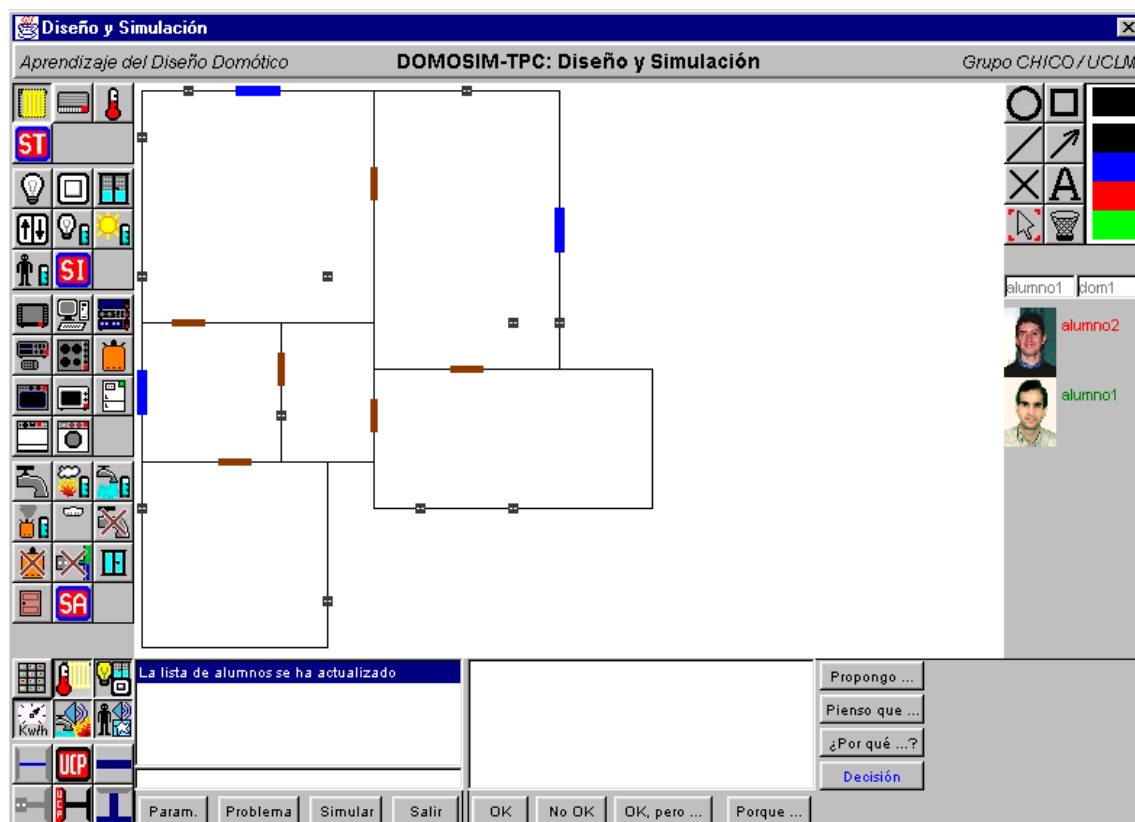
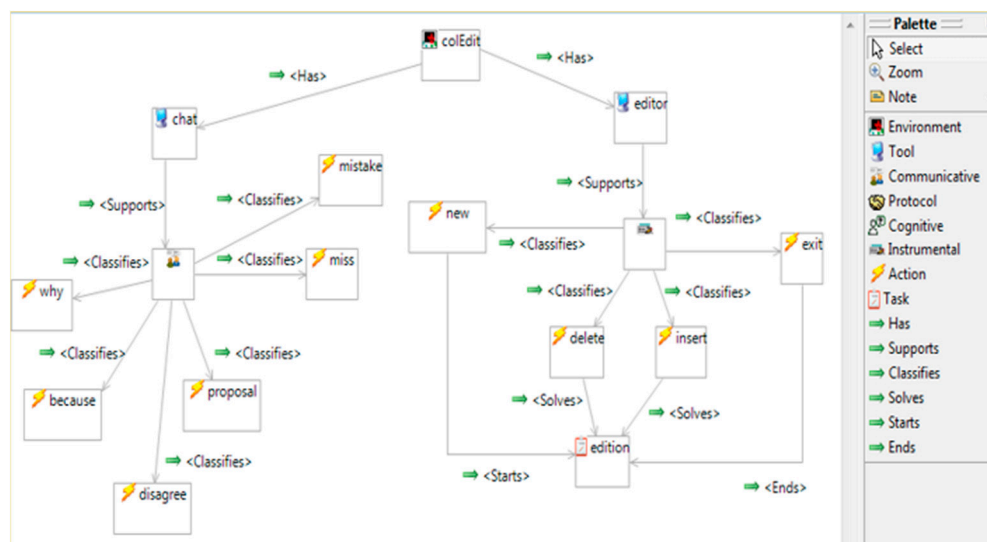


Figure 2. DomoSim-TPC synchronous CSCL environment.

Students must solve home automation problems of increasing complexity, and the system uses artificial intelligence-based mechanisms to check the quality of the proposed result with an aid implemented through scaffolding. As mentioned above, the collaboration process and the quality of the solution proposed by the students are monitored on the basis of a set of qualitative and quantitative variables expressed by means of linguistic variables based on fuzzy logic. To obtain these conclusions, we use variables modeled with fuzzy sets. These sets are compound attributes or linguistic labels which define the domain of each variable. We considered variables characterized by five specific attributes corresponding to the following labels: very low, low, normal, high, and very high. In DomoSim-TPC, we incorporated the necessary functionality to draw conclusions about initiative, creativity, elaboration, conformity and disconformity.

The process-product analysis in CSCL environments carried out in the DomoSim-TPC environment is very important in any collaborative system. The quality of the proposed solution—i.e., the product—and the quality of the collaboration—i.e., whether a fruitful collaboration between the participants has taken place—must be known. Finally, it is also important to know whether the quality of the collaboration has influenced the quality of the product. To this end, a line of research in collaboration analysis was developed. It allows detailed reports of the collaborative activity to be obtained using a framework called

FAPPEC (Framework for Process-Product Analysis in Collaborative Environments) [9]. This framework can be used in different tools to monitor the collaborative activity of the participants in a collaborative environment and can be defined by means of a tool with direct manipulation of the different parameters to be visualized. In Figure 3, we can see this authoring tool designed to model the intervention of a CSCL system.



**Figure 3.** Authoring tool for intervention modelling.

Through these frameworks, precise metrics can measure the collaboration of students to solve complex problems. The metrics used come from studies of the influence of different parameters upon the collaborative process [8–10]. Some of these metrics for the collaborative process are:

1. Number of accesses to the system that each student carried out to work in the activity.
2. Number and mean of contributions made.
3. Mean of the contributions' size.
4. Kind of contributions made.
5. Number of replied and refined contributions.
6. Depth of the discussion for each task of the problem.
7. Number of times the last news was accessed.
8. Number of messages sent (classified by kind: planning, coordination, or system).
9. Number of instantaneous messages sent.

### 3. Ubiquitous Computing

Ubiquitous computing [11] is an interaction paradigm that describes different devices which collaborate through networks facilitating, in a non-intrusive way, the tasks that a user wants to perform. These devices are located wherever they are needed, hence the name “ubiquitous computing”, meaning anywhere.

An example of systems that can be encompassed within the paradigm of ubiquitous computing is the AULA (A Ubiquitous Language Appliance) system [12]. The AULA system was designed to teach foreign languages using personal digital assistants (PDAs). The students have to produce a collaborative piece of writing by dividing it in a structured way into ideas and aspects. The system can be used in or out of the classroom, and there is a synchronization mechanism for the generated papers.

AULA has a web-based version called A Web-based Language Appliance (AWLA) [13], which has been successfully used in language courses and is complemented by the AIOLE (An Interactive Online Learning Environment) system, a web-based system generator to learn foreign languages.

AWLA (A Web Language Appliance) (Figure 4) is a wiki-like system with various online language tools that allows the collaborative editing of texts in different languages with teacher correction possibilities and tools to prevent plagiarism by the learners. The whole system can be classified as a personal learning environment (PLE) with the capability to be used within the classroom, anytime and anywhere according to the paradigm of ubiquitous computing, through the use of PDAs.

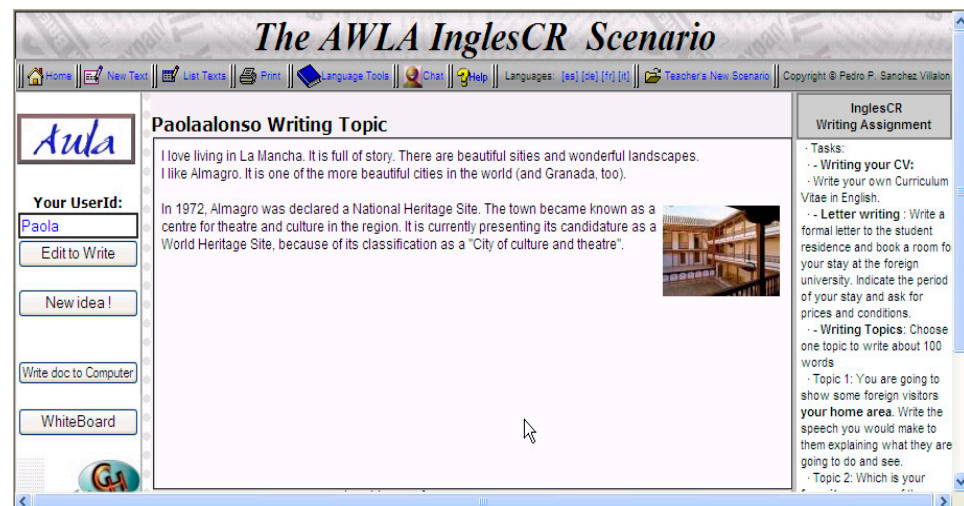


Figure 4. AWLA and AIOLE personal learning environment.

#### 4. Groupware Engineering

The ad hoc implementation of collaborative systems is an arduous task. This is why the automatic or semi-automatic creation of groupware applications is useful for the generalization of their use in classroom. The automatic creation of synchronous applications for both CSCL and Computer Supported Cooperative Work (CSCW) environments is vital to developing useful tools in a reasonable time. In this case, a model-driven engineering (MDE) method has been designed for the development of domain-independent collaborative modelling systems [10]. Figure 5 shows an application generated using the SPACE-DESIGN tool in order to allow the participants (with their picture on the right of the system in F) to generate diagrams in a collaborative way. Figure 5 also shows a shared whiteboard (A) where two participants can add elements from the application domain, in this case logic gates. Entities (B) and relationships (C) can be added and appear on the left in the environment (D, E). The telepointers (G) are used to show the area of the shared whiteboard where each participant is editing. This environment has been developed by defining models that automatically generate the final environment in Figure 5.

In order to model collaborative and interactive environments, notations and methodologies are necessary to help describe the systems that will be developed, as well as to automate the processes that generate the collaborative tools. Among the existing proposals, we highlight ConcurTaskTree (CTT) [14] and Collaborative Interactive Application Methodology (CIAM) [15].

CIAM [15] is a methodology that guides the designer to create a conceptual specification of the aspects to be considered within a groupware system in order to architect the system and design the necessary interaction in this kind of groupware system. The different steps to be carried out to obtain the final user interface can be seen in Figure 6.



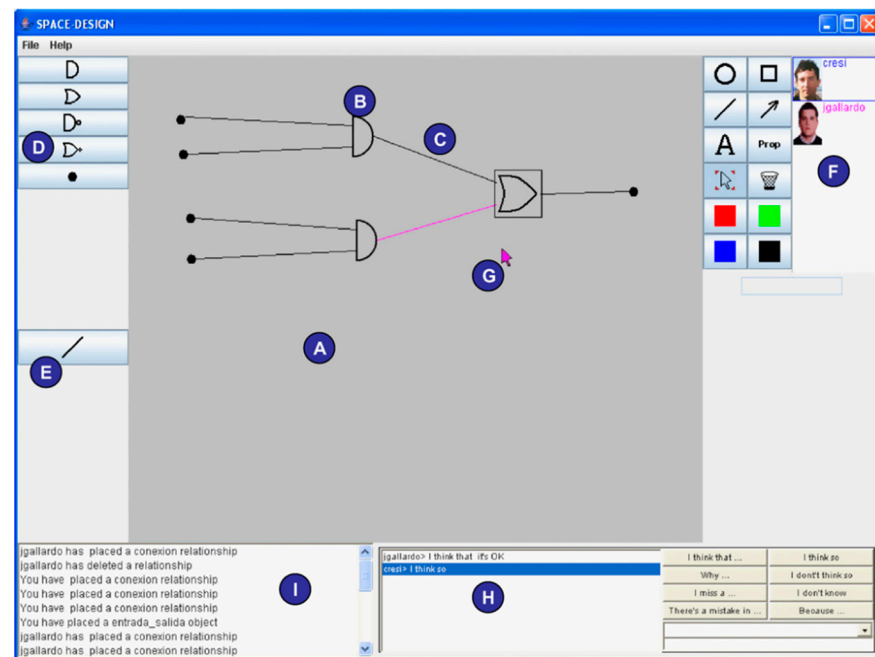


Figure 5. Collaborative environment generated by models.

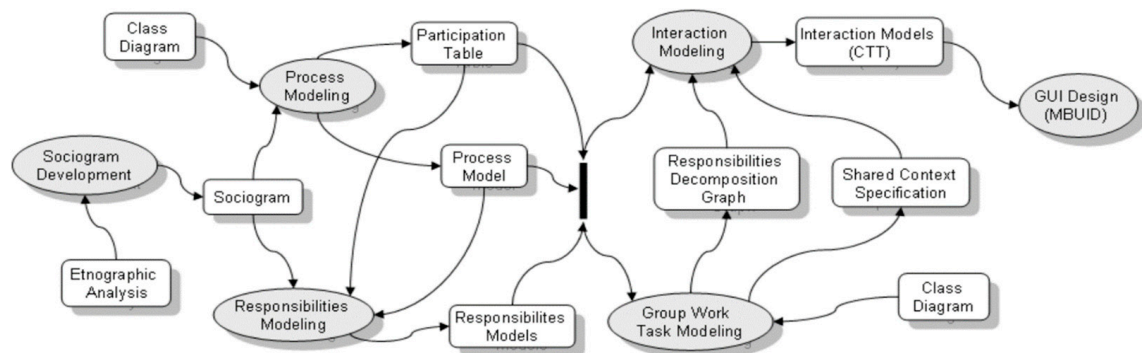


Figure 6. The CIAM proposal and the products obtained in each of its stages.

CIAM (Collaborative and Interactive Application Methodology) is complemented by a notation called CIAN (Collaborative Interactive Application Notation) [15]. A summary of the CIAN is shown in Figure 7. In areas A and B, we can see the icons that represent the nodes that constitute the process model, the relationship types which indicate the different tasks, and the interdependence types. In C, we can see the icons used to represent an interaction task model in CTT notation.

The CIAN design produces a CTT diagram that in turn generates user interfaces in a semi-automatic way. To facilitate this design, a tool called CIAT (Collaborative Interactive Application Tool) [16] is used. With this tool, the whole methodological process can be followed up to the generation by means of model-based user interface design (MBUID).

The CIAN was extended to increase expressiveness in the development of e-Learning applications with the Learn-CIAN expansion and its corresponding Learn-CIAT tool [17], which can be seen in Figure 8.

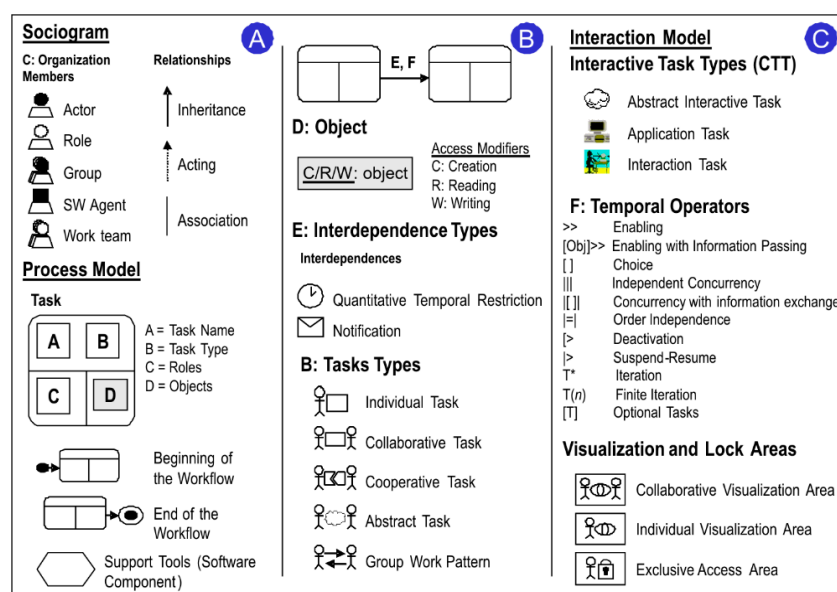


Figure 7. The CIAN notation.

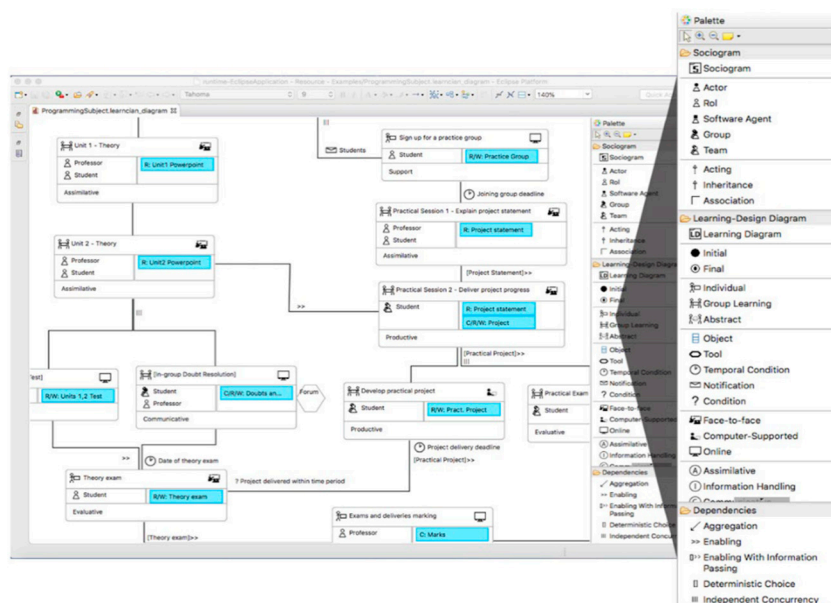


Figure 8. A learning design specified using the Learn-CIAT tool.

## 5. Usability and User Experience

All the developed systems require usability studies to demonstrate their efficiency in carrying out the tasks that the user or student, as the case may be, wants to perform.

MoLEF (Mobile Learning Evaluation Framework) [18] is a proposed framework in which both the usability and the pedagogical characteristics of an e-Learning environment can be evaluated with a large number of metrics. Figure 9 shows the graphical representations of the technological and pedagogical usability parameters of a mobile application developed by the CHICO group. MoLEF has been used as a basis together with CIAM in the development of Learn-CIAN, which we discussed in the previous chapter.

Classical usability studies based on questionnaires have been completed in recent years using more objective and direct sources of information, among which we highlight the use of eye tracking techniques. The concept of eye tracking refers to a set of technologies that make it possible to record and analyze the way a person looks at a given image (or user interface), providing physiological information on aspects related to the interest, attention,

and cognitive effort involved in the visual analysis of the information displayed on the screen [19].

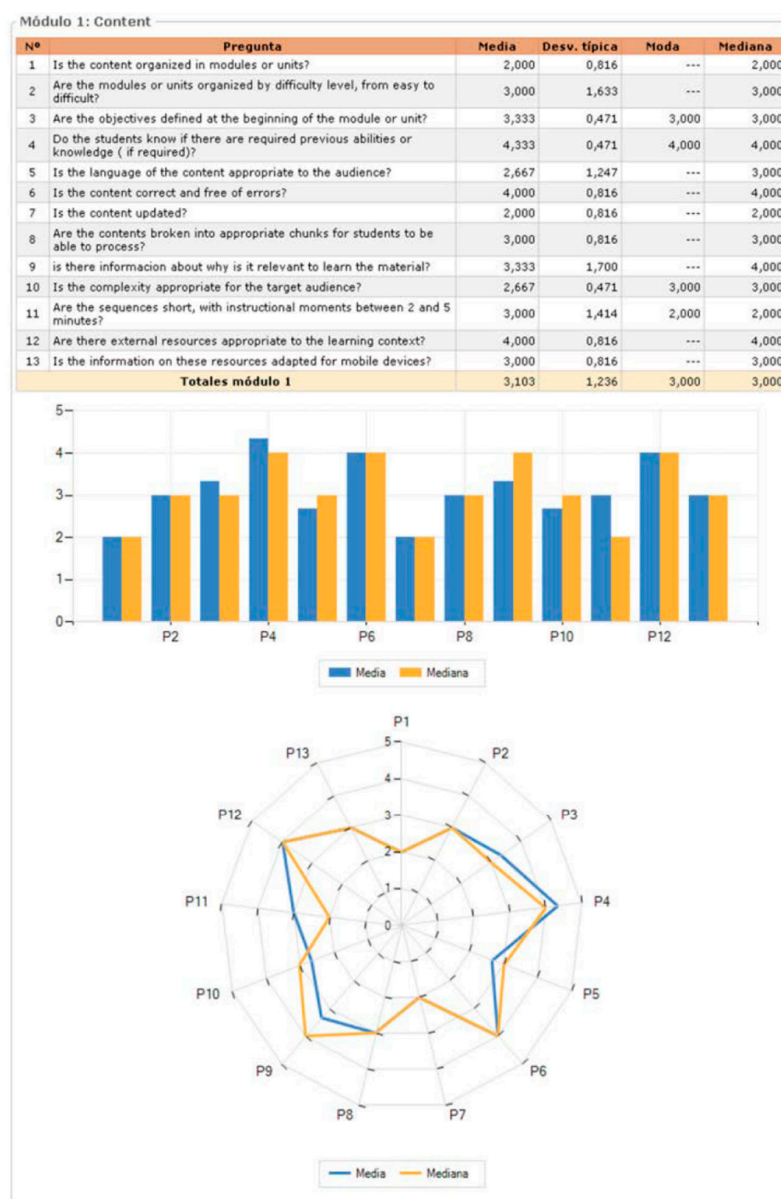


Figure 9. A screenshot of the tool implementing the MoLEF framework.

This eye tracking technique can be used in the evaluation of educational software in the field of programming teaching and, in particular, to teach greedy algorithms [20]. This technique has also been used to evaluate awareness characteristics in groupware systems [21] used in the CIAM [22].

The use of eye tracking in the study of multimedia educational materials is a very promising field. In particular, it has been very effective in studies aimed at primary school students, in the field of mathematics teaching [23,24].

The use of eye tracking techniques has been complemented in all the case studies with classical system usability tests. In the MoLEF [18], questionnaires with Likert-type scales are intensively and extensively proposed. The set of applications from Greedex to Greedex-Tab 2.0 [20] is an example of how different user interfaces are improved by system usability tests.



In all the studies, qualitative questionnaires with a Likert scale are used to improve the usability and productivity of the proposed systems [21]. In order to measure the usability and productivity of the CIAM, usability tests were carried out among software engineers comparing these tests with other methodologies through performance of different exercises. Once the exercises were completed by the subjects, a qualitative questionnaire was provided. Each subject was required to answer some questions related to the ease of use, the perceived complexity of each model, the suitability and usefulness of the notation, and specific aspects related to some of the CIAN diagrams. Additionally, some questions about intention to use were provided. With these results, we concluded that CIAN is a notation that has proven useful to help software engineers in the modeling process of interactive and collaborative aspects in groupware applications.

The usability and productivity of different tools were also evaluated by the students, by means of questionnaires in refs. [25,26]. For the Cole Programming system [25], we used usability tests answered by the students on the different collaborative tools used. Analyzing the results, we observed how the use of forums and voting pools seems less widespread when scoring these tools.

The conclusion drawn from the use of eye tracking is that objective usability measures complement and improve classical usability analysis techniques and are of great help in the area of human–computer interaction.

## 6. A Case Study: Learning Environments for Programming

As an example, we will discuss learning environments for programming. Learning to program can be a difficult challenge for students in computer science. For this reason, systems based on artificial intelligence have been built in order to develop intelligent tutoring systems (ITS) such as the Cole-Programming system and COALA (Computer Assisted Environment for Learning Algorithms) [25,26], which uses a collaborative system to teach programming on top of the integrated development environment (IDE), Eclipse [27].

Cole-Programming and COALA (Figure 10) allow the definition of a possible solution to an algorithm, and through fuzzy logic, the programming environment, implemented as an Eclipse plugin, compares the solution given by the student with the one proposed by the teacher and helps in solving the programming problems. The environment is also collaborative and allows the addition of written comments via pen-based interaction to the students' solution.

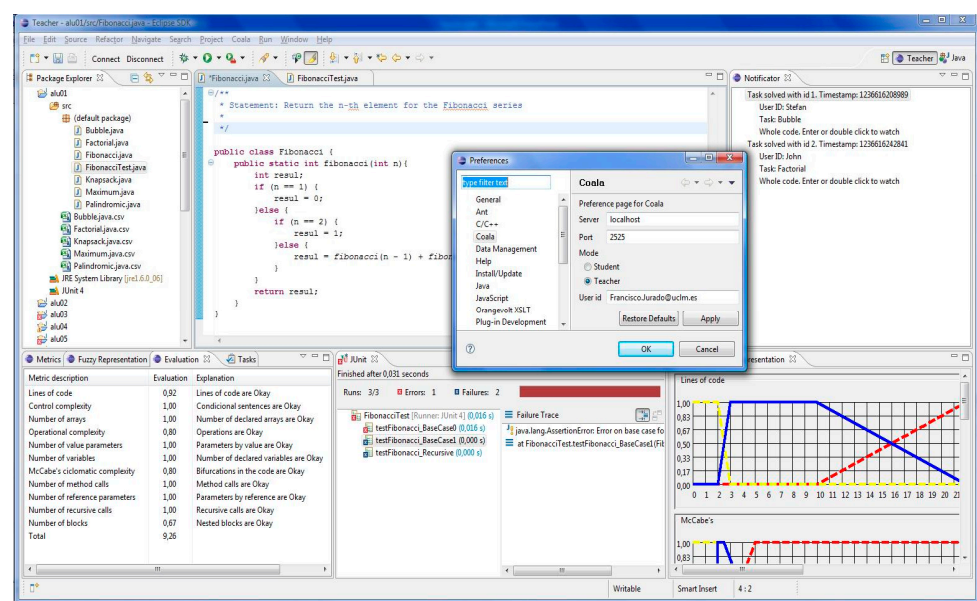


Figure 10. Cole-Programming over the Eclipse IDE (Integrated Development Environment).

COLLECE (COLLAborative Edition, Compilation and Execution of programs) [28] is another 100% collaborative teaching environment (Figure 11) implemented on top of the Eclipse IDE. It is complemented by a notation (ANGELA, notAtioN of road siGns to facilitatE the Learning of progrAmming) [29] that allows the visualization (Figure 12) of algorithms through a metaphor of roads and signals, which in turn allows its use to teach programming in the early stages of the students' training. It is important to note that it can not only be utilized to teach programming, but also in peer programming environments. In addition, it encourages the creation of groups of programmers collaborating to solve complex problems.

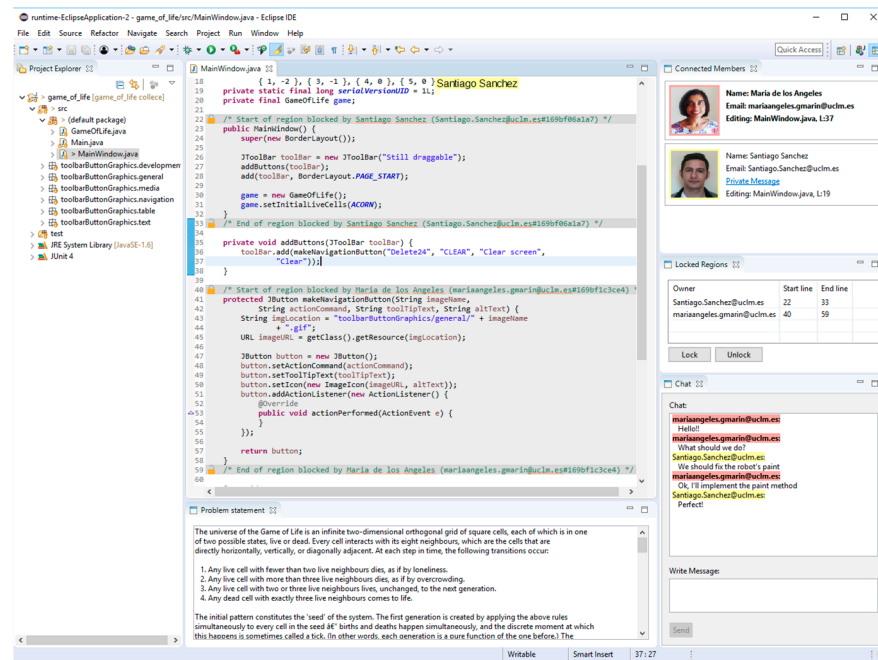


Figure 11. COLLECE 2.0 as a plugin for Eclipse IDE.

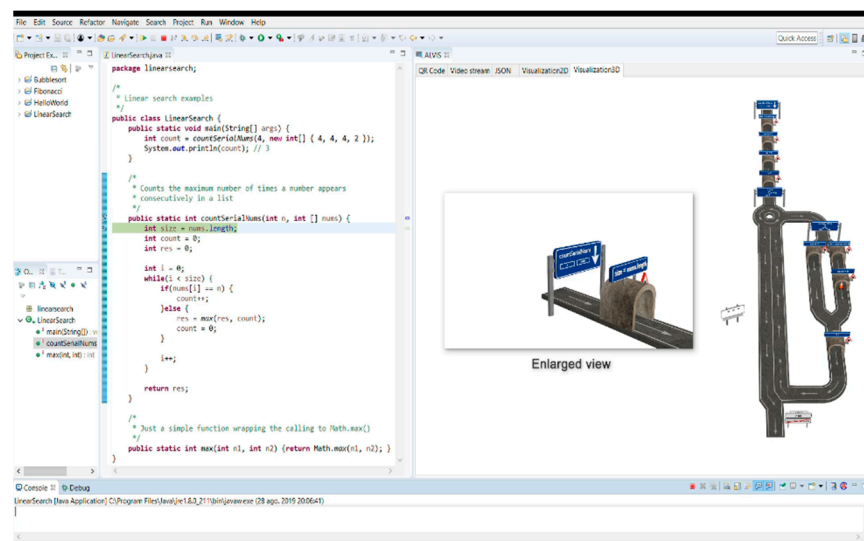


Figure 12. ANGELA visual plugin for Eclipse IDE.

The system described can visualize the algorithms in ANGELA notation using augmented reality Microsoft HoloLens glasses and interact with the visualized algorithm (Figure 13). This feature is highly valued by the students who can interact with the system even while debugging the code within a visual environment.

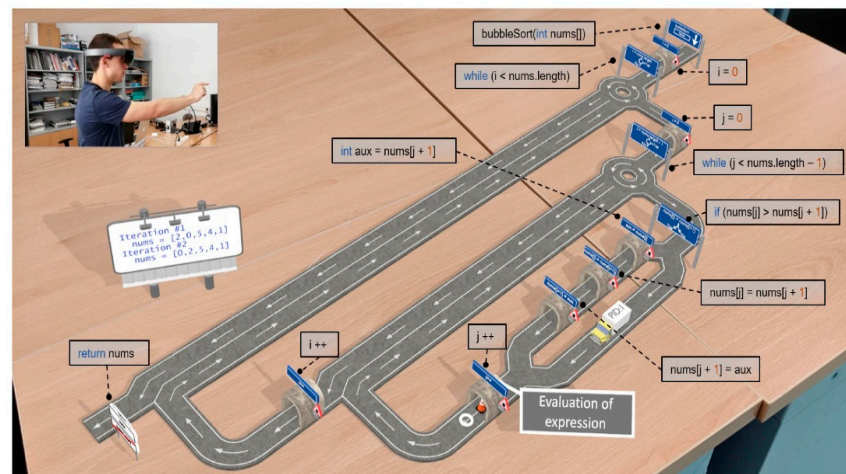


Figure 13. ANGELA visualization with an augmented reality HoloLens device.

## 7. Conclusions

The following series of conclusions from our work can be used as a guide in the development of e-Learning systems with more capabilities than the current systems and that provide help in learning when a large part of the students' work is carried out online.

Currently, the prevailing paradigm in online environments is the use of Learning Management Systems (LMS) or Massive Open Online Courses (MOOC), where the content is hosted in presentations or texts, and to a lesser extent with explanatory videos of certain parts of the course. Synchronous collaborative systems have also started to be used on a regular basis, especially due to the ongoing COVID-19 pandemic. A survey on educational process mining considering MOOCs, LMSs, and their relationships with collaborative systems, as well as a summary and tables relating these systems, can be found in reference [30].

However, the first recommendation of this article is the use of asynchronous collaborative systems to plan solutions to complex problems using artificial intelligence. These planning systems, and the others that will be described, need to be included within LMSs in an easy way, and for this purpose LMSs must allow the use of plugins that insert the necessary features for 21st century learning.

Scaffolding should be implemented for complex problems in the LMS. The proposed problems will be easier at the beginning, assisted by AI-based systems and will become more difficult as the students complete successive problems, whereas the help will diminish until it disappears. In this way, the students will improve their knowledge of the problems addressed.

Synchronous collaborative environments will allow the problems to be solved by several students once the problems have been planned in asynchronous collaborative systems. Again, it is vital that these systems be integrated into LMSs easily and efficiently. This is one of the results of "The Next Generation Digital Learning Environment. A Report on Research" [31]. According to Brown et al., "What is clear is that the LMS has been highly successful in enabling the *administration* of learning but less so in enabling learning itself". This report argues that the Next Generation Digital Learning Environment (NGDLE) must address five domains of core functionality:

- Interoperability and integration;
- Personalisation;
- Analytics, advice, and learning assessment;
- Collaboration;
- Accessibility and universal design.

In this report, the authors explain that "The support for collaboration must be a *lead* design goal, not an afterthought. The current LMS is often designed on the transmission

model of education—a mechanism to transmit syllabi, content, and assessments.” According to Lin et al., [32] a combination of a web-based collaborative problem-solving system and teacher guidance can be implemented to develop students’ collaborative problem-solving skills.

It is also important that we have frameworks which measure the degree of student collaboration and the quality of the answers to the problems faced, and which interrelate the two measures. In other words, we can check whether a good collaboration has produced a good solution to the proposed problem. Again, these frameworks should be integrated into LMS or MOOCs.

Creating tools that support collaborative problem solving is not feasible if they have to be implemented ad hoc. Hence, model-based user interface design (MBUID) and model-driven engineering (MDE) are needed to support the efficient development of these learning tools. Methodologies and notations specific to e-Learning such as Learn-CIAM or general ones such as CIAM-CIAN should help to define interactive and collaborative learning environments.

The user experience and usability of the proposed systems and their objective measurement are also important, which is why eye tracking systems are necessary in the development of educational tools.

Finally, we must take into account all the new interaction paradigms, such as virtual reality, augmented reality, or ubiquitous computing, so that our e-Learning tools provide real and effective learning.

In conclusion, this article presents a series of guidelines for the use of e-Learning systems that can help design future systems based on the experience of a research group in HCI and e-Learning of more than 20 years.

It is found that although the first intelligent tutor systems and collaborative systems are mature technologies, they have not been introduced conveniently in the current systems. However, their efficient use is proposed by different authors.

**Funding:** This research was funded by the Ministry of Economy, Industry and Competitiveness, and the European Grant Number “TIN2015-66731-C2-2-R”.

**Data Availability Statement:** No new data were created or analyzed in this study. Data sharing is not applicable to this article.

**Acknowledgments:** Most of the projects presented in this paper were developed within the CHICO group [3] over the last 25 years. Until 2020, the author was the director of this research group. The authorship of the papers corresponds to the authors cited in the corresponding articles. The author thanks the various co-authors of the cited papers for their contribution to the developments presented. In particular, he would like to thank Pedro P. Sánchez for his important contribution to the CHICO research group.

**Conflicts of Interest:** The author declares no conflict of interest.

## References

1. Ortega, M. *Computers in Education: The Near Future, in Computers and Education in the 21st Century*; Ortega, M., Bravo, J., Eds.; Kluwer Academic Publishers: Dordrecht, The Netherlands; Boston, MA, USA, 2000; pp. 3–16. [\[CrossRef\]](#)
2. Di Pietro, G.; Biagi, F.; Costa, P.; Karpiński, Z.; Mazza, J. *The Likely Impact of Covid-19 on Education: Reflections Based on the Existing Literature and Recent International Datasets*; Publications Office of the European Union: Luxembourg City, Luxembourg, 2020. [\[CrossRef\]](#)
3. CHICO Group. 2021. Available online: <http://blog.uclm.es/grupochico/> (accessed on 7 January 2021).
4. Redondo, M.A.; Bravo, C.; Ortega, M.; Verdejo, M.F. Providing adaptation and guidance for design learning by problem solving. The DomoSim-TPC approach. *Comput. Educ.* **2007**, *48*, 642–657. [\[CrossRef\]](#)
5. Redondo, M.Á.; Bravo, C. DomoSim-TPC: Collaborative Problem Solving to Support the Learning of Domotical Design. *Comput. Appl. Eng. Educ.* **2006**, *4*, 9–19. [\[CrossRef\]](#)
6. Bravo, C.; Redondo, M.Á.; Ortega, M.; Verdejo, M.F. Collaborative environments for the learning of design: A model and a case study in Domotics. *Comput. Educ.* **2006**, *46*, 152–173. [\[CrossRef\]](#)
7. Bravo, C.; Redondo, M.Á.; Ortega, M.; Verdejo, M.F. Collaborative Distributed Environments for Learning Design Tasks by Means of Modelling and Simulation. *J. Netw. Comput. Appl.* **2006**, *29*, 321–342. [\[CrossRef\]](#)



8. Collazos, C.A.; Gutiérrez, F.L.; Gallardo, J.; Ortega, M.; Fardoun, H.M.; Molina, A.I. Descriptive theory of awareness for groupware development. *J. Ambient Intell. Humaniz. Comput.* **2019**, *10*, 4789–4818. [\[CrossRef\]](#)
9. Duque, R.; Bravo, C.; Ortega, M. A model-based framework to automate the analysis of users' activity in collaborative systems. *J. Netw. Comput. Appl.* **2011**, *34*, 1200–1209. [\[CrossRef\]](#)
10. Duque, R.; Bravo, C.; Ortega, M. An ontological approach to automating collaboration and interaction analysis in groupware systems. *Knowl. Based Syst.* **2013**, *37*, 211–229. [\[CrossRef\]](#)
11. Weiser, M. The Computer for the 21st Century. *Sci. Am.* **1991**, *265*, 66–75. [\[CrossRef\]](#)
12. Paredes, M.; Sánchez-Villalón, P.P.; Ortega, M.; Velázquez-Iturbide, J.Á. Collaborative Composition in a Foreign Language with Handheld Computing and Web Tools. *J. Univers. Comput. Sci.* **2007**, *13*, 948–958. [\[CrossRef\]](#)
13. Villalon, P.P.S.; Ortega, M. AWLA and AIOLE for Personal Learning Environments. *Int. J. Contin. Eng. Educ. Life-Long Learn.* **2007**, *17*, 418–431. [\[CrossRef\]](#)
14. Paternò, F. ConcurTaskTrees: An Engineered Notation for Task Models. In *The Handbook of Task Analysis for HCI*; Diaper, D., Stanton, N.A., Eds.; LEA: Mahwah, NJ, USA, 2004; pp. 483–501.
15. Molina, A.I.; Redondo, M.Á.; Ortega, M. A Methodological Approach for User Interface Development of Collaborative Applications: A case study. *Sci. Comput. Program.* **2009**, *74*, 754–776. [\[CrossRef\]](#)
16. Molina, A.I.; Giraldo, W.J.; Gallardo, J.; Redondo, M.A.; Ortega, M.; García, G. CIAT-GUI: A MDE-Compliant Environment for Developing Graphical User Interfaces of Information Systems. *J. Adv. Eng. Softw.* **2012**, *52*, 10–29. [\[CrossRef\]](#)
17. Molina, A.I.; Arroyo, Y.; Lacave, C.; Redondo, M.A. Learn-CIAN: A visual language for the modelling of group learning processes. *Br. J. Educ. Technol.* **2018**, *49*, 1096–1112. [\[CrossRef\]](#)
18. Navarro, C.X.; Molina, A.I.; Redondo, M.A. Towards a model for evaluating the Usability of m-Learning systems: From a mapping study to an approach. *IEEE Lat. Am. Trans.* **2015**, *13*, 552–559. [\[CrossRef\]](#)
19. Bojko, A. *Eye Tracking the User Experience: A Practical Guide to Research*; Rosenfeld Media: New York, NY, USA, 2013.
20. Arroyo, Y.; Molina, A.I.; Lacave, C.; Redondo, M.A.; Ortega, M. The GreedEx experience: Evolution of different versions for the learning of greedy algorithms. *Comput. Appl. Eng. Educ.* **2018**, *26*, 1306–1317. [\[CrossRef\]](#)
21. Molina, A.I.; Gallardo, J.; Redondo, M.A.; Ortega, M.; Giraldo, W.J.; Orozco, W.J.G. Metamodel-Driven Definition of a Visual Modeling Language for Specifying Interactive Groupware Applications: An empirical study. *J. Syst. Softw.* **2013**, *86*, 1772–1789. [\[CrossRef\]](#)
22. Molina, A.I.; Redondo, M.A.; Ortega, M.; Hoppe, U. CIAM: A Methodology for the Development of Groupware User Interfaces. *J. Univers. Comput. Sci.* **2008**, *14*, 1434–1446.
23. Navarro, O.; Molina, A.I.; Lacruz, M.; Ortega, M. Evaluation of multimedia educational materials using eye tracking. *Procedia Soc. Behav. Sci.* **2015**, *197*, 2236–2243. [\[CrossRef\]](#)
24. Molina, A.I.; Navarro, O.; Ortega, M.; Lacruz, M. Evaluating Multimedia Learning Materials in Primary Education using Eye Tracking. *Comput. Stand. Interfaces* **2018**, *59*, 45–60. [\[CrossRef\]](#)
25. Jurado, F.; Molina, A.I.; Redondo, M.A.; Ortega, M. Cole-Programming: Shaping Collaborative Learning Support in Eclipse. *IEEE-Rita* **2013**, *8*, 153–162. [\[CrossRef\]](#)
26. Jurado, F.; Redondo, M.A.; Ortega, M. eLearning Standards and Automatic Assessment in a Distributed Eclipse Based Environment for Learning Computer Programming. *J. Comput. Appl. Eng. Educ.* **2014**, *22*, 774–787. [\[CrossRef\]](#)
27. Eclipse. 2021. Available online: <https://www.eclipse.org/> (accessed on 7 January 2021).
28. Schez-Sobrino, S.; Gmez-Portes, C.; Vallejo, D.; Glez-Morcillo, C.; Redondo, M.Á. An Intelligent Tutoring System to Facilitate the Learning of Programming through the Usage of Dynamic Graphic Visualizations. *Appl. Sci.* **2020**, *10*, 1518. [\[CrossRef\]](#)
29. Schez-Sobrino, S.; García, M.Á.; Lacave, C.; Molina, A.I.; Glez-Morcillo, C.; Vallejo, D.; Redondo, M.Á. A modern approach to supporting program visualization: From a 2D notation to 3D representations using augmented reality. *Multimed. Tools Appl.* **2020**, *80*, 543–574. [\[CrossRef\]](#)
30. Romero, C.; Ventura, S. Educational data mining and learning analytics: An updated survey. *Wires Data Min. Knowl. Discov.* **2020**, *10*, e1355. [\[CrossRef\]](#)
31. Brown, M.; Dehoney, J.; Millichap, N. The Next Generation Digital Learning Environment. EDUCAUSE, 2015. Available online: <https://library.educause.edu/resources/2015/4/the-next-generation-digital-learning-environment-a-report-on-research> (accessed on 7 January 2021).
32. Lin, K.Y.; Yu, K.C.; Hsiao, H.S.; Chang, Y.S.; Chien, Y.H. Effects of web-based versus classroom-based STEM learning environments on the development of collaborative problem-solving skills in junior high school students. *Int. J. Technol. Des. Educ.* **2020**, *30*, 21–34. [\[CrossRef\]](#)