A review of webapp authoring tools for e-learning

M. Latorre[1], A. Robles-Gómez[2], L. Rodríguez[3], P. Orduña[3], E. San Cristóbal[1], A. C. Caminero[2], Ll. Tobarra[2], I. Lequerica[1], S. Ros[2], R. Hernández[2], M. Castro[1], D. Lopez-de-Ipiña[3], J. García-Zubia[4]

[1] Electrical and Computer Engineering Department, UNED, Spain
{mlatorre,elio,ilequerica,mcastro}@ieec.uned.es
[2] Control and Communication System Department, UNED, Spain
{arobles,accaminero,llanos,sros,roberto}@scc.uned.es
[3] DeustoTech – Deusto Institute of Technology, University of Deusto, Spain
{luis.rodriguezgil,pablo.orduna,dipina}@deusto.es
[4] Faculty of Engineering, University of Deusto, Spain
zubia@deusto.es

Abstract: The lack of tracking and storing capabilities for the results of web-based learning activities is an issue that remains unsolved. Transitions or interactions defined by teachers through a set of conditions still require programming skills that stay far beyond the desired final results. In addition to this, authoring tools should be powerful enough to let lecturers generate contents which are high-quality, interactive, and tuned to each student's cognitive preferences and progress. Availability and processing capabilities, or motivation, relevance, etc., must also be aspects to address in this context. For these reasons, this paper aims to review the existing web application authoring toolkits focusing on distance education. In particular, we analize their main features, requirements and issues, as well as the most promising areas for future improvemenst in this field.

Keywords-webapps; authoring tools; e-learning; distance education; customization;

I. INTRODUCTION

Educational-oriented applications are a breakthrough in the context of inquiry-based learning/e-learning. The diversity of software in mobile and tablet platforms has been accomplished through a set of authoring tools and open standards that enable expert users and developers to edit, package, and publish web-based programs on several domains, being Engineering one of the most prominent ones. Teachers are encouraged to select, create and customize their resources (e.g. simulations, step-by-step exercises, interactive animations, etc.) tailored to a specific learning scenario and, also, share them with just few clicks. However, most of the software available nowadays can be difficult to configure and deploy in a computer for the members of the academic community, and its usage can also be excessively time-consuming. This way, teachers would devote more time to these issues than to the creation of high quality contents and to the adaptation of them to the learning necessities of their students. This is especially true in the context of the European Higher Education Area (EHEA) [1], where the focus is placed in the continuous evaluation of the students' progress.

In our particular case, the Spanish University for Distance Education (UNED, *Universidad Nacional de Educación a Distancia*) is a distance university which is devoted to provide university training to students who lack the time or resources needed to attend on-campus classes. This is necessary for students who have tight timetable because of other commitments (e.g. work or family commitments), thus fitting on-campus classes into their timetable becomes unfeasible. Furthermore, UNED provides university training to convicts and sick people, whose attendance to on-campus classes is not possible. UNED provides such students with the platform to receive education at home without the time-consuming and sometimes money-consuming effort requested by on-campus universities.

The evolution of education and the increase in the knowledge necessities of our society requires have created significant changes with regard to the way how the learning process takes place. Nowadays, there is a constant need to improve, to keep our knowledge up-to-date or to obtain knowledge on new topics – this being especially true in the case of Engineering, where technology is constantly evolving. Distance education is a solution to this problem, since it allows students to obtain practical knowledge without the space and time constraints of classical face-to-face education – thus allowing them to fit their studies into their possibly tight schedules.

In numbers, UNED is the largest university in Spain in terms of number of students (more than 200,000) and number of lecturers (more than 1,500). UNED employs around 6,900 local tutors, who are spread over 61 associated centers all over Spain. Furthermore, UNED also provides university training to more than 2,100 students in more than 11 countries. Recall that interactions between students and university are by means of the technological infrastructures of the university, that is, students download material and upload essays and other works through the web site, read and send emails, and read and post messages in the forums.

Therefore, it can be seen the tremendous necessities of our University with regard to the distant access to the teaching resources, in which these resources are a key. This highlights our need to develop web-app authoring tools to let our lecturers

generate learning contents that have (1) high-quality, (2) interactive, and (3) that are customized to the each student's cognitive preferences and progress. Availability and processing capabilities, along with motivation, or relevance, are also issues to be addressed. According to this, the main contribution of this work is an analysis of the existing web application toolkits. In particular, their main features, requirements and issues will be explored in detail to provide a general overview of their current model of work and areas for future improvement.

The structure of this paper is as follows: Section II describes the motivations for our work in terms of educational web-based toolkits. After that, Section III details all the components and/or categories of web-app authoring tools (logic programming systems, visual block editors, app packaging generation systems, and interface generators and editors). Next, the most relevant specialized editors for the creation of virtual and remote laboratories are presented in Section IV. Finally, Section V presents conclusions and suggests guidelines for future work.

II. EDUCATIONAL WEB-BASED TOOLKITS

The evolution of standards for Web application development has made possible a wide array of online editors and management systems for remote sites. Proprietary and free software platforms based on collections of graphical controls or libraries of predesigned elements represent a simple method to compose a static document with a few interactive elements (e.g. audio or video) without editing the source code. Nevertheless, static documents have been replaced with interactive environments which provide a set of features and functionalities almost identical to their locally-installed counterparts.

Teachers are able to explain precise concepts through interactive simulations, generally known as apps, with the help of slideshow tools or more advanced toolkits (traditionally Flash or Java). The resulting apps (no external installation will be needed) and the store infrastructure attract sufficient awareness and usage by third-party developers and users in a higher scale in comparison with specialized systems, such as institutional repositories [2]. Android marketplace is an example of popular ranking site to store these apps making it visible to a large audience [3]. If an authoring toolkit is interested in reaching the educational community, it must also provide a list of characteristics that must be interesting for teachers. Hence, as a starting point, there are relevant aspects in the design of apps focused on learners of different ages. They can be divided into the following categories:

- Logical systems used by teachers for sequencing activities (from PowerPoint-like content to dynamic simulations).
- Visual block editors for widescreen devices and mobile devices.
- App packaging and generation systems.
- Interface generators and editors.

Although several destination platforms (LMS, PLE, others) exist, a variety of interface technology configurations are employed (such as HTML5, Mash-up, single page apps, native-apps, mobile apps, etc.), many standard formats and APIs…, all of them share a core options to contain the same functionality. All these elements are the foundations for further exploration. For this reason, the different components of a web-app authoring tool will be examined in the next section.

III. COMPONENTS OF AN APP AUTHORING TOOL

A. *Logic programming systems (solely used in e-learning)*

In order to guarantee that any logical system provides the support that teachers demand for their classes, we need to formulate the following question: *how can we test if a system of this nature works effectively?* Under that condition, one of the most important aspects that this language representation must include among its characteristics is its ease of use.

End-users with different skills have a dependency factor on the scientific notation, which is required to write a program from scratch [4]. Four types of notation can be tailored to novice programmers: textual, iconic, visual or tangible (i.e. physical objects). As a consequence, researchers explore different methods to simplify the process of creating software through a friendly interface [5]. A large diversity of environments implement one of these notations, or a combination of them. Two goals can be achieved here: teach students how to program, or build interactive learning resources.

On one hand, school teachers and non-expert programmers make use of alternative systems for writing pure source code in a text editor. It is through them that instructors are able to show how a program can be expressed in natural language or how it evolves in the course of time (e.g. steps or events of a known process). One of the first breakthroughs on this topic is the visual environment "Alice 3D". Students can create "scenes" through visual tools: interactions between elements, scripting with blocks, etc., see [6-10]. PsyScope also belongs to this group [11] [12]. Advanced graphical solutions based on sketches and objects emerged to provide further improvements over these tools:

- IPRO, a social and mobile gaming path to programming learning [13].
- Greenfoot, a highly graphical integrated development environment (IDE) for learning object oriented programming [14].
- Donburi, social game creation for non-coders [15].
- Toontalk, programming with a videogame [16].
- Netlogo [17].
- Sketch-speech interface to teach concurrent programming [18].
- E-Block, a tangible programming tool with physical objects [19].

Due to the massive usage of mobile devices in all their forms, related works about user experience deal with their application in software. On that regard, Android has been tested as a possible environment for learning to code [20]. There have also been some efforts to measure the application of the AppInventor mobile development software in Computer Science [21].

Aside from the previous examples, learning to program software by playing is an approach that has been taken by different projects. For instance, Light-bot is an educational app for mobile phones with an innovative user interface [22]. Other games are used to teach programming, such as Cargo-bot (recursion), Code Avengers, PiktoMir (preschoolers), Code Hero or Meta!Blast [23]. Other approaches have been taken into account by instructors in order to provide a simple way to understand conditional structures and constraints. For instance, Squeak, which is based on Smalltalk, the first language in which everything is built with objects.

The creation of interactive games for students of primary schools, such as the one mentioned before, demands a considerable effort from the teacher side but the experiences are greatly improved [24]. Scratch, an evolution of LOGO, is one of the most extended systems that let learners connect different blocks with a particular functionality to build a complete program or animation [25]. This 2-D visual programming language is one of the most known choices to teach students of primary schools. Teachers can develop games and instructional content with this free and open source tool from the Massachusetts Institute of Technology (MIT). The combination of existing elements (e.g. images, audio files) is quite straightforward. Other languages may be used to structure the classroom exercises. Furthermore, it is possible to collaborate with learners from different ages or levels [26]. Even as the context of use is focused on programming, these tools are not tied to a single scientific domain. For instance, there are methodologies and informal assessments of its success applied in Mathematics [27]. Another tool for authoring serious games similar to Scratch is e-Adventure [28].

Graphical representations are also a suitable resource to teach programming. In fact, languages as the state stage transition command graph (GRAFCET), a mode of representation and analysis of industrial automation systems, are best suited to follow this path. These kind of supporting techniques get a high grade of satisfaction from learners that applied them in their learning flow [29]. Nevertheless, a deeper evaluation is still required to properly assess the benefits and workarounds for the "code blocks" modes of representation [30] [31].

On the other hand, learning paths are a widely used technique to structure e-learning content. From the lecturers' perspective, sequencing online activities could be considered a form of implementing the program logic in the different steps required to pass a learning path.

Visual modeling environments for instructional design represent a relevant step towards this direction: store the conditions to go forward/backward in an exercise with a standard format [32]. Teachers encode that information with educational modeling languages [33]. However, the complexity of writing and validating the rules in structured formats (XML is the de-facto format) becomes a barrier for instructors, even with the help of specialized editors [34]. IMS Learning design is the e-learning standard on that regard. Interoperability between IMS LD and the Unified Modeling Language (UML) shows the similarities in these two representations [35]. Units of learning, the basic component of a learning design, are edited with authoring tools such as the ReLoad LD Editor or ReCourse [36] [37]. The translation of these descriptions into other formal designs is problematic [38].

The main conclusion extracted from experiences carried out on this field of study is that visual technologies improve the students' learning process [39]. However, there exists a gap to be filled in this context: simple authoring tools for learning sequences that do not require programming skills. As a workaround to the previous challenge, it is necessary to mention that authoring interactions and transitions is not restricted to the e-learning field. Professional mockup builders include a form-based input to add events into a set of slides, such as the exposed by many app prototyping tools (Figure 1). Content creators for e-learning may benefit from this direct method that increases the interactivity feature in their current projects.

B. Visual block editor

If there is one context where visual blocks (diagrams, hierarchies, classes, or flowcharts) have been used, extensively in early stages of development, then this is software programming. Several data flow visual programming languages emerged since the 1970's, LabView among them [40]. Academic purposes, design models, factory automation (e.g. Petri nets and Kahn's process networks), or optimization of business processes are some areas where these languages have gained a wide adoption. Madgets are a variant applied in the world of interactive tabletops.

Although the technology behind web apps is completely different to the previous contexts, service-oriented architectures have also embraced this model. Mash-ups are also the final result of many platforms. The University of Groningen (Netherlands) has Apache Rave in production for its "My University" portal which gives access to 45,000 users [41].

Nowadays, graphical blocks, handwriting and tactile input are common methods to arrange components in computer-based systems. The design of the user interface (UI) or presentation of apps can be intuitively done taking advantage of the drag and drop capabilities offered by tablets. Although UI controls or widgets are the building blocks, the workspace where these components are shown takes control of the final layout.

The preferred web workspace where teachers send their activities takes control of the global behavior. Wikis, Learning Management Systems (LMS) or any custom e-learning systems impose restrictions in the content that can be visualized and browsed: structure, packaging, file formats, and aggregation. There are two groups responsible of this restriction by matter of security and consistency of the final styles: LMS and Personal Learning Environments (PLE).

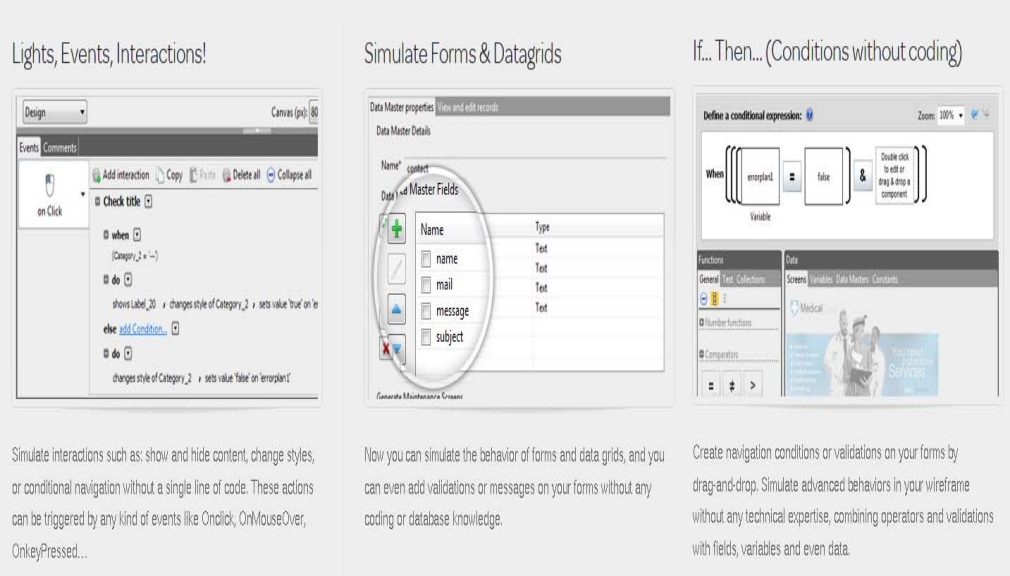

Fig.1 Example of a graphical interaction editor.

Learning Content Management Systems (LCMS) are the first group of interest. As their mainstream counterparts (CMS such as WordPress), an LCMS lets designers of web pages use other users' templates and load small blocks of content (e.g. tabs, search, or categories) to customize the complete site. They are also used to add features and content in a flexible way without any need for coding. Plugins are a more powerful solution to the previous technology that allows users to add and move around widget areas and control what widgets display and when to do so [42].

The most adaptable solution to date for learners are the PLEs, which are composed by widgets. According to this, these are composed by a set of nominal apps. The pedagogical and statistical analysis of these e-learning platforms shows improvements in the learning process [43]. Scenarios have been proposed in the literature [44]. Recent research has focused on the composition of PLEs with institutional resources and external widgets. For instance, evaluation of the usefulness and usability of this platform has been carried out in computer science courses. This type of system can be considered as the next generation learning platform, so its features are included in other systems [45].

In the context of connecting sensors and control from a Personal Computer (PC), we can find the concept of "phidgets". These components are a set of user friendly building blocks that let developers simply plug in the device and write a program using practically any language [46]. With PhidgetLab is possible to access several phidgets from a Squeak image. On the other hand, LogiSketch makes use of a new technique called "hover widgets" for increasing the capabilities of pen-based interfaces. It is useful for sketching logic circuits (gates) in a tablet environment [47]. CircuitLab is another simulator with a vast library of pre-designed blocks for drawing electrical circuits.

According to this, software that is available only as a traditional desktop application is unlikely to receive substantial adoption among cross-platform users. As a matter of fact, the number of mobile apps has grown exponentially. However, mobile devices such as smart phones or tablets impose higher restrictions for the development of software in comparison with personal computers or laptops. The size of their screens in the former case goes from 10" to smaller ones. Speed is also a concern, due to the difficult balance between processing power and longer life of the batteries (even as current products hold quad processors). Users of these devices cannot be in disadvantage when handling a laboratory. Responsive designs have answered to these requirements.

Most integrated development environments use simulators to test the final results. However, the advent of tablets has opened the door to the authoring of apps through a tactile-based platform. This section also addresses the development of laboratory experiments with devices that have small screens and the existing (if any) editors to do so.

Only a short list of applications has been published that justify the feasibility of programming with touch screen-based devices such as smart phones or tablets. Although there is not an operating system of choice in the mobile ecosystem, the open source system Android has taken a prominent place in the last decade. For instance, the Android IDE (AIDE) oriented to devices with this operating system [48]. Catroid is also an on-device graphical programming language for Android devices developed by the Lifelong Kindergarten Group at the MIT Media Lab that is inspired by the Scratch programming language [49]. Proprietary development platforms oriented to the web design with HTML5 set of standards for small screens can be found, where the Sencha Touch framework is a good example.

On the other hand, TouchDevelop targets both students and hobbyists, because the apps written with this editor can personalize the mobile phone [50]. Another tool worth mentioning is Codea for Apple iPad. Interactive creations that use the internal features of these ranges of tablets (cameras,

multi-touch or accelerometers) are possible with this code editor [51]. Therefore, it is mainly geared towards games. As a last option, we can find the extensions (collaboration mechanisms, user interface adaptations, event processing and performance measuring) to the web-based development environment Lively kernel [52].

C. App generation and packaging system

The composition of web apps with external services has reached the next step of their evolution with the third generation of service oriented architectures (SOA). Research studies have demonstrated that users are capable of generating visual applications consisting of software services (also known as graphical components).

WIDGAT [53] represents the first online "code free" app creation tool for teachers and students. These tools offer a better mechanism to interact with the web than other rigid infrastructures, such as pure client-central server configurations. Another noteworthy tool for composing applications from simple components is the web mash-up editor MyCocktail, which was developed in the context of the EU FP7 project Romulus4 [54].

On the other hand, these applications are stored on a specific repository. Apache Wookie is one of the servers that have served as a test platform in order to verify the feasibility of these blocks of information in PLEs. Some scenarios have been carried out to know how they compare in certain areas of application. Other social platforms, such as iGoogle, Netlog or Yandex apps, integrated the complete lifecycle process in the web site.

IMS Common Cartridge (IMS CC) is another alternative for building apps for a full-fledged PLE with modularity in mind. Nevertheless, interoperability becomes a key issue for future development within the field of e-learning when dealing with these technologies.

D. Interface generators and editors

Given the high dependency that the available software suffers from physical configurations and their components (i.e. input/output transducers) for which it was originally designed, most developers can only rely on the proprietary tools provided by manufacturers. LabView and ELVIS are the most popular platforms for remote experiments [55], but other software tools fulfill the needs demanded by teachers for virtual experiment design or more simple scenarios. There is a short selection of integrated development environments (IDE) and services, which will be described in next section, that provide the means required to design the user interface of a laboratory. These programs include collections of icons, buttons and their associated events for interacting with a device connected to Internet. Aside from these generic tools, there are applications designed for a specific scientific field.

IV. Specialized Editors (Virtual and Remote Laboratory Creation)

There is a short selection of IDEs and services that provide the required means to design the user interface of a resource for experiments: virtual and remote web laboratories. These programs include collections of icons, buttons and their associated events for interacting with a device connected to Internet.

For instance, the remote lab designed by the Grammar-school of J. Vrchlicky (GymKT) makes use of the K8055-MARIE management application [56]. Another software of interest is the Internet School Experimental System (iSES), with 450 installations deployed across the Czech Republic and Slovak Republic [57]. Cloudduino is a project to make accessible any Arduino-based experiment with an attractive web dashboard [58]. A similar platform for implementing experiments with a collection of sensors and the documentation of its interfaces of communication is Gadgeteer. As well as the tools to program its Arduino counterpart, the Gadgeteer .NET UI Designer is a rapid software development environment for prototyping embedded applications [59]. Another framework tailored specifically to online simulations is LateNiteLabs, a commercial software that allows developers to code the complete laboratory and physical tools with Flash in a way that resembles to the real workspace [60].

Aside from these generic tools, there are applications designed for a more specific scientific field. The most relevant ones [61-74] are listed in Table I with their respective categories.

From the extensive list that is included in Table I, the open source ROS [73] and MoveIt! [74] stand out for the characteristics of our particular field of research. Briefly, ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. On the other hand, the MoveIt! tool is a software targeted at allowing to build advanced applications, which integrates motion planning, kinematics, collision checking with grasping, manipulation, navigation, perception, and control.

V. Conclusions and Future Work

The development of tracking and storing capabilities for the results of web-based learning activities is still an unresolved

TABLE I. APPPLICATIONS DESIGNED FOR A SPECIFIC SCIENTIFIC DISCIPLINE.

Discipline	Applications
Chemistry	ChemCollective [61]
Computer Science	SimQuest [62], The SoftLab Experience [63], SoftLab-Vlabs [64], Virtual Intelligent SoftLab [65]
Physics	Interactive Physics [66], Newton Playground [67], Algodoo [68]
Mathematics	swMath [69], Cinderella [70], GeoGebra [71]
Thermodynamics	Armfield HT15 [72]
Robotics	ROS [73], MoveIt! [74]

issue. Teachers may define or need transitions between learning states which may require programming skills that stay far beyond the desired final results. Authoring collaborative experiences in the current platforms is a troublesome task that relies on external and unsecure tools from the institutions.

In order to improve the learning process of students compared to traditional approaches, slideshows must be converted into learning paths with different levels of difficulty. Authoring tools have to be powerful enough to let lecturers generate high-quality interactive content, and also they must allow the customization of the learning contents to the each student's cognitive preferences and progress. Given the growing number of available features that web technologies have in new versions, this goal is a step closer to be completed. The flexibility achieved in current publishing platforms, LMS and PLE alike, paves the way for e-learning oriented web-apps. Availability and processing capabilities, or motivation, and relevance, are aspects to address in this context. However, the performed studies included only a small number of participants, so it is an open field for further research.

On the other hand, it can be seen the tremendous necessities of our University (UNED, the largest University in Spain) with regard to the distant access to the teaching resources, in which these web-apps are a key. Therefore, this work provides an analysis of the existing web application toolkits. Their main features, requirements and issues have been explored in detail to provide a general overview of their current model of work and areas for future improvement. From this study, we have been able to choose the most relevant features. These features are: open/proprietary tool, content quality, presentation design, method of creation (manual, automatic, or semi-automatic), enrichment method (adding further information in a course), (previous) experience of authors, separation of presentation and logic, multi-device support, metadata generation, internationalization, level of interaction, accessibility, reusability, ease of use, environment of publication (LMS, PLE, platform independent: web sites, wikis, etc.), standards compliance (IMS Common Cartridge, SCORM 1.2, SCORM 2004, AICC PENS, xAPI, HTML5, etc), tracking, and assessment. Currently, we will compare the most relevant webapp authoring tools analyzed in this work, in a similar way to [75]. After that, we plan to develop a web-app authoring tool to let lecturers generate high-quality interactive content, which can be customized to each student's cognitive preferences and progress. Availability and processing capabilities, along with motivation, and relevance, among other features, will also be taken into account in the framework of this development.

ACKNOWLEDGMENTS

Authors would like to acknowledge the support of the following European Union projects: RIPLECS (517836-LLP-1-2011-1-ES-ERASMUS-ESMO), PAC (517742-LLP-1-2011-1-BG-ERASMUS-ECUE), EMTM (2011-1-PL1-LEO05-19883), and MUREE (530332-TEMPUS-1-2012-1-JO-TEMPUS-JPCR). Furthermore, we also thank the Community of Madrid for the support of E-Madrid Network of Excellence (S2009/TIC-1650).

REFERENCES

[1] The European Higher Education Area, Joint Declaration of the European Ministers of Education, Bologna Declaration 1999. Available in: http://www.eua.be/eua/jsp/en/upload/OFFDOC_BP_bologna_declaration.1068714825768.pdf. Date of last access: 9th November, 2013.

[2] C. Ullrich, R. Shen, and K. Borau, "Learning from Learning Objects and Their Repositories to Create Sustainable Educational App Environments," IEEE 13th International Conference on Advanced Learning Technologies (ICALT), 2013, pp.285-287.

[3] J. Ludwig, R. Richards, B. Presnell, and D. Fu, "A General Framework for Developing Training Apps on Android Devices," In The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC), National Training Systems Association, 2012.

[4] A. MacLean, K. Carter, L. Lôvstrand, and T. Moran, "User-Tailorable Systems: Pressing the Issues with Buttons," In Human Factors in Computing Systems: CHI'90 Conference Proceedings (Seattle, WA, USA), 1990, pp. 175-182.

[5] C. Kelleher, and R. Pausch, "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers," ACM Computing Surveys (CSUR), vol. 37, no. 2, 2005, pp. 83-137.

[6] Alice publications. Available in: http://www.alice.org/index.php?page=publications/publications. Date of last access: 9th November, 2013.

[7] M. Conway, S. Audia, T. Burnette, D. Cosgrove, and K. Christiansen, "Alice: lessons learned from building a 3D system for novices," In Proceedings of the SIGCHI conference on Human factors in computing systems, 2000, pp. 486-493, ACM.

[8] W. P. Dann, S. Cooper, and R. Pausch, Learning to Program with Alice (w/CD ROM), 2011, Prentice Hall Press.

[9] D. S. Tan, G. G. Robertson, and M. Czerwinski, "Exploring 3D navigation: combining speed-coupled flying with orbiting," In Proceedings of the SIGCHI conference on Human factors in computing systems, 2001, pp. 418-425, ACM.

[10] S. H. Rodger, J. Hayes, G. Lezin, H. Qin, D. Nelson, R. Tucker, et al. "Engaging middle school teachers and students with alice in a diverse set of subjects," In ACM SIGCSE Bulletin, vol. 41, no. 1, 2009, pp. 271-275, ACM.

[11] B. Macwhinney, J. Cohen, and J. Provost, "The PsyScope experiment-building system. Spatial vision," vol. 11, no. 1, 1997, pp. 99-101.

[12] J. L. Borton, M. A. Oakes, M. E. Van Wyk, and T. A. Zink, "Using PsyScope to conduct IAT experiments on Macintosh computers," Behavior research methods, vol. 39, no. 4, 2007, pp. 789-796.

[13] T. Martin, M. Berland, T. Benton, and C. P. Smith, "Learning Programming with IPRO: The Effects of a Mobile Social Programming Environment," Journal of Interactive Learning Research, vol. 24, no. 3, 2013, pp. 301-328.

[14] M. Kölling, "The greenfoot programming environment," ACM Transactions on Computing Education (TOCE), vol. 10, no. 4, 2010, p. 14.

[15] J. Kim, E. Chen, R. Mittal, and G. Roth, "Donburi: Social Game Creation for Non-coders", 2012.

[16] K. Kahn, "ToonTalk: An Animated Programming Environment for Children," Journal of Visual Languages & Computing, vol. 7, no. 2, 1996, pp. 197-217.

[17] S. Tisue, and U. Wilensky, "NetLogo: A simple environment for modeling complexity," In International Conference on Complex Systems, 2004, pp. 16-21.

[18] D. Schlegel, A Sketch/Speech Interface and Usability Study for Software to Teach and Learn Concurrent Programming, 2009.

[19] D. Wang, Y. Zhang, and S. Chen "E-Block: A Tangible Programming Tool with Graphical Blocks," Mathematical Problems in Engineering, 2013.

[20] T. A. Nguyen, S. T. Rumee, C. Csallner, and N. Tillmann, "An experiment in developing small mobile phone applications comparing on-phone to off-phone development," In User Evaluation for Software Engineering Researchers (USER), 2012, pp. 9-12, IEEE.

[21] App Inventor, Massachusetts Institute of Technology (MIT) (2012). Available in: http://appinventor.mit.edu/. Date of last access: 9th November, 2013.

[22] L. A. Gouws, K. Bradshaw, and P. Wentworth, "Computational thinking in educational activities: an evaluation of the educational game light-bot," In Proceedings of the 18th ACM conference on Innovation and technology in computer science education, 2013, pp. 10-15, ACM.

[23] M. E. Stenerson, Academic game development: practices and design strategies for creating STEM games, 2012.

[24] Á. del Blanco, J. Torrente, E. J. Marchiori, I. Martínez-Ortiz, P. Moreno-Ger, and B. Fernández-Manjón, "A Framework for Simplifying Educator Tasks Related to the Integration of Games in the Learning Flow," Educational Technology & Society, vol. 15, no. 4, 2012, pp. 305-318.

[25] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, et al., "Scratch: programming for all," Communications of the ACM, vol. 52, no.11, 2009, pp. 60-67.

[26] A. Wilson, T. Hainey, and T. Connolly, Evaluation of Computer Games Developed by Primary School Children to Gauge Understanding of Programming Concepts, In proceedings of the 6th European Conference on Games Based Learning (ECGBL), October 4-5, 2012, pp. 549-558.

[27] An Investigation into the Use of Scratch to Teach KS3 Mathematics. Available in: http://scratched.media.mit.edu/sites/default/files/Scratch%20Dissertation%20Complete.pdf. Date of last access: 9th November, 2013.

[28] J. Torrente, P. Moreno-Ger, B. Fernández-Manjón, and J. L. Sierra, "Instructor-oriented authoring tools for educational videogames," In 8th IEEE International Conference on Advanced Learning Technologies (ICALT), 2008, pp. 516-518, IEEE.

[29] R. L. Marichal, and E. J. González, "ULLSIMGRAF: An educational tool with syntax control for Grafcet notation," Computer Applications in Engineering Education, 2012.

[30] Evaluation of Scratch Software for the Development of Teaching Materials to Promote Student - Centered Leaning in Primary Schools in Mongolia. (2012). Available in: http://www.yamaguchi.gsic.titech.ac.jp/wp-content/uploads/downloads/2013/02/2012_M_Yano_Mid-TermAbstract.pdf. Date of last access: 9th November, 2013.

[31] Y. Shotaro, "Development of teaching material for primary school in Mongolia: Lessons from teachers feedback," ICT4D, 56th Annual Conference Comparative and International Education Society (CIES), April 22-27, 2012.

[32] M. Derntl, and R. Motschnig-Pitrik, "coUML: A Visual Language for Modeling Cooperative Environments," In L. Botturi, & T. Stubbs (Eds.), Handbook of Visual Languages for Instructional Design: Theories and Practices, 2008, pp. 154-182, Hershey, PA: Information Science Reference. doi:10.4018/978-1-59904-729-4.ch009.

[33] J. Torres, C. Cirdenas, J. M. Dodero, and E. Juirez, Educational Modelling Languages and Service Oriented Learning Process Engines, 2010, Mary Beth Rosson eds., doi: 10.5772/8236.

[34] R. Queir, Seqins-A Sequencing Tool for Educational Resources. In Proceedings of 2nd Symposium on Languages, Applications and Technologies, SLATE 2013, June 20-21, pp. 83-96.

[35] R. Barchino, J. R. Hilera, L. De-Marcos, J. M. Gutiérrez, et al., Interoperability between visual UML design applications and authoring tools for learning design, International Journal of Innovative Computing, Information and Control, 2012, vol. 8, no.2.

[36] I. Martinez-Ortiz, J. L., Sierra, and B. Fernandez-Manjon, Authoring and reengineering of IMS learning design units of learning, IEEE Transactions on Learning Technologies, 2009, vol. 2, no. 3, pp. 189-202.

[37] A. Mavroudi, and T. Hadzilacos, Implementation of adaptive learning designs.

[38] I. Martínez-Ortiz, J. L. Sierra, and B. Fernández-Manjón, Translating e-learning Flow-Oriented Activity Sequencing Descriptions into Rule-based Designs, In Information Technology: New Generations (ITNG), 2009, pp. 1108-1113, IEEE.

[39] M. Karakus, S. Uludag, E. Guler, S. W. Turner, and A. Ugur, Teaching computing and programming fundamentals via App Inventor for Android, In Information Technology Based Higher Education and Training (ITHET), 2012, pp. 1-8, IEEE.

[40] M. Marttila-Kontio, "Visual Data Flow Programming Languages," Publications of the University of Eastern Finland, Dissertations in Forestry and Natural Sciences, no. 30, 2011.

[41] Integrating Social Apps with Content Driven Sites using Apache Rave and Spring HMVC. Presentation slides available at https://http://es.slideshare.net/ate.douma/apache-coneu2012-t142. Date of last access, October 1st, 2013.

[42] S. Wilson, P. Sharples, and D. Griffiths, "Distributing education services to personal and institutional systems using Widgets," 1st International Workshop on Mashup Personal Learning Environments (MUPPLE08), 2008, pp. 25-32.

[43] F. J. García Peñalvo, M. Á. Conde González, and M. J. Rodríguez Conde, Percepción de la apertura de los LMS en las ramas educativas y tecnológicas, 2012.

[44] M. Á. Conde, F. J. Garcia-Penalvo, and M. Alier, Interoperability scenarios to measure informal learning carried out in PLEs. In Intelligent Networking and Collaborative Systems (INCoS), 2011, pp. 801-806, IEEE.

[45] S. Ros, R. Hernández, A. Robles-Gómez, A. C. Caminero, L. Tobarra, and E. S. Ruíz, Open Service-Oriented Platforms for Personal Learning Environments. IEEE Internet Computing, vol. 17, no. 4, 2013, pp. 26-31.

[46] S. Greenberg, C. and Fitchett, Phidgets: easy development of physical interfaces through physical widgets, In Proceedings of the 14th annual ACM symposium on User interface software and technology, 2001, pp. 209-218, ACM.

[47] C. Alvarado, Sketch recognition for digital circuit design in the classroom, In 2007 Invited Workshop on Pen-Centric Computing Research.

[48] L. F. Hernández Ramírez, F. A. Giraldo Giraldo, and Fundación Universitaria San Martín, Programación gráfica sobre dispositivos Android, 2013.

[49] W. Slany, "Catroid: a mobile visual programming system for children," In Proceedings of the 11th International Conference on Interaction Design and Children, 2012, pp. 300-303, ACM.

[50] N. Tillmann, M. Moskal, J. de Halleux, and M. Fahndrich, "Touchdevelop: Programming cloud-connected mobile devices via touchscreen," In Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software, 2011, pp. 49-60, ACM.

[51] M. Hesenius, C. D. O. Medina, and D. Herzberg, "Touching factor: software development on tablets," In Software Composition, 2012, pp. 148-161, Springer Berlin Heidelberg.

[52] C. Calmez, H. Hesse, B. Siegmund, S. Stamm, A. Thomschke, et al., Explorative authoring of Active Web content in a mobile environment, 2013, ISSN:1613-5652. Available in: http://pub.ub.uni-potsdam.de/volltexte/2013/6405/. Date of last access: 9th November, 2013.

[53] WIDGAT – Widget Design Authoring Toolkit. Available in: http://arc.tees.ac.uk/widgat/. Date of last access: 9th November, 2013.

[54] C. A. Iglesias, J. I. Fernández-Villamor, D. Del Pozo, L. Garulli, and B. García, Combining domain-driven design and mashups for service development, In Service Engineering, 2011, pp. 171-200, Springer Vienna.

[55] J. Del Rio, E. Trullols, A. Manuel, J. Mendez, G. Prados, and R. Palomera, ELVIS. A new tool for teaching and training. In Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference (IMTC), 2004, vol. 2, pp. 1282-1285.

[56] N. Wagner, Velleman K8055 Geratetreiber, 2009.

[57] F. Schauer, F. Lustig, J. Dvořák, and M. Ožvoldová, "An easy-to-build remote laboratory with data transfer using the Internet School Experimental System," European Journal of Physics, vol. 29, no. 4, 2008, pp. 753.

[58] Cloudduino - Arduino Remote Control and Monitoring from the Cloud. (2013). Available in: http://sourceforge.net/projects/labscript/. Date of last access: 9th November, 2013.

[59] N. Villar, J. Scott, and S. Hodges, “Prototyping with microsoft. net gadgeteer,” In Proceedings of the 5th international conference on Tangible, embedded, and embodied interaction, 2011, pp. 377-380, ACM.

[60] K. Pyatt, and R. Sims, “Virtual and physical experimentation in inquiry-based science labs: attitudes, performance and access,” Journal of Science Education and Technology, vol. 21, no. 1, 2012, pp. 133-147.

[61] D. Yaron, M. Karabinos, D. Lange, J. G. Greeno, and G. Leinhardt, “The ChemCollective—virtual labs for introductory chemistry courses,” Science, vol. 328, no. 5978, 2010, pp. 584-585.

[62] W. R. Joolingen and T. Jong, “SimQuest, authoring educational simulations,” Authoring Tools for Advanced Technology Learning Environments: Toward cost-effective adaptive, interactive, and intelligent educational software, 2003, pp. 1-31.

[63] A. C. Catlin, M. G. Gaitatzes, E. N. Houstis, Z. Ma, S. Markus, et al., “The SoftLab experience: Building virtual laboratories for computational science,” 1995. Availale in: http://www.cs.purdue.edu/research/cse/softlab/softlab-vlabs/softlab-framework/softlab_report/report.html. Date of last access: 9th November, 2013.

[64] E. S. S. Aziz, S. K. Esche, and C. Chassapis, “Content rich interactive online laboratory systems,” Computer Applications in Engineering Education, vol. 17, no.1, 2009, pp. 61-79.

[65] B. Y. Kathane and P. B. Dahikar, “Design and Implementation of Adder and Subtractor Experiments using Virtual Intelligent SoftLab,” International Journal of Computer Science and Telecommunications (IJCST), vol. 3, no. 1, 2012.

[66] A. Jimoyiannis and V. Komis, “Computer simulations in physics teaching and learning: a case study on students' understanding of trajectory motion,” Computers & education, vol. 36, no. 2, 2001. pp. 183-204.

[67] M. Ventura, V. Shute, and Y. J. Kim, “Assessment and Learning of Qualitative Physics in Newton’s Playground,” Artificial Intelligence in Education, 2013, pp. 579-582, Springer.

[68] Le, C. H. E. N, “Research and Practice of Using Algodoo in High School General Technology Course,” Modern Educational Technology, vol. 11, no. 21, 2010.

[69] swMath (2011). Available in: http://www.swmath.org/. Date of last access: 9th November, 2013.

[70] Cinderella (2013). Available in: http://cinderella.de/tiki-index.php. Date of last access: 9th November, 2013.

[71] M. Hohenwarter, and J. Preiner, “Dynamic mathematics with GeoGebra,” Journal of Online Mathematics and its Applications, vol. 7, 2007.

[72] A. Hyder, S. K. Choi, and D. Schaefer, “Remotely controlled laboratory experiments: Creation and examples,” Systems and Information Engineering Design Symposium (SIEDS), 2010, pp. 62-67.

[73] M. Quigley, K. Conley, B. Gerkey, J. Faust, et al., “ROS: an open-source Robot Operating System,” ICRA workshop on open source software, vol. 3, no. 3.2, 2009.

[74] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), vol. 3, 2004, pp. 2149-2154.

[75] B. Nikolic, Z. Radivojevic, J. Djordjevic, and V. Milutinovic, “A survey and evaluation of simulators suitable for teaching courses in computer architecture and organization,” IEEE Transactions on Education, vol. 52, no. 4, pp. 449 –458, nov. 2009.