OpenSocial Application Builder and Customizer for School Teachers

Luis Rodriguez-Gil*, Miguel Latorre†, Pablo Orduña*, Antonio Robles-Gómez†, Elio Sancristobal†,
Sten Govaerts‡, Denis Gillet‡, Irene Lequerica†, Agustin C. Caminero†, Roberto Hernandez†,
Salvador Ros†, Manuel Castro†, Diego López-de-Ipiña* and Javier Garcia-Zubia*
*DeustoTech – Deusto Institute of Technology
University of Deusto, Bilbao, Spain
Email: {luis.rodriguezgil,pablo.orduna,dipina,zubia}@deusto.es
†Spanish University for Distance Education, UNED, Spain
Email: {mlatorre,elio,ilequerica,mcastro}@ieec.uned.es, {arobles,accaminero,roberto,sros}@scc.uned.es
‡École Polytechnique Fédérale de Lausanne, EPFL, Switzerland
Email: {sten.govaerts,denis.gillet}@epfl.ch

***Abstract*—**

Nowadays, a large number of online laboratories are available and deployed throughout the world. Most of them are sponsored by universities and often only used by their home institution, because they tend to be dedicated to the needs of their creators. However, in many cases these labs provide successful experiences and other teachers would wish to incorporate them into their classroom activities and to embrace Learning-by-Experience methodologies.

Currently, they cannot do that effectively without first tailoring the lab experience to their own teaching style and the educational background of their students. This can be a complicated affair, due the closed nature of many online labs and the steep learning curve of the few tools that allow customization. This paper describes the work on an application builder, named *App Composer*, which aims to make it easy for school teachers to create their own customized applications from existing ones.

Different customization levels are provided, each one powered by a different type of composer. For instance, teachers will be able to easily translate an application into a new language, or to start with a template and create a customized version of an application. The App Composer makes use of Graasp and OpenSocial – a widely-used, interoperable and open technology.

***Keywords*-K-12 education; application builder; distance education**

I. Introduction

In recent years, remote and virtual laboratories have been developed and grown in importance. Many institutions use them already –specially universities– but they very rarely share them. This leads to the existence of many underused laboratories with nonetheless great potential.

School teachers could often make great use of remote and virtual laboratories but they generally lack the resources and technical expertise. In these cases, they consider using laboratories that other institutions provide, but frequently these labs are not suited for their specific needs. To integrate them into a course, they would need to adapt them to their preferences, teaching styles, syllabus and student level.

In this context, the European Go-Lab Project (Global Online Science Labs for Inquiry Learning at School)[1] aims to make online remote and virtual laboratories more accessible to be used on a large scale in education, by providing a technical framework through which teachers can enrich their classroom activities with the experiences and demonstrations that online laboratories enable. The Go-Lab Project consists of several parts. This work will focus on an application builder, known as the *App Composer*. The purpose of the App Composer is to enable school teachers to develop their own customized applications using existing ones as a basis.

Within the Go-Lab project, teachers are able to create learning spaces for students consisting of online labs and inquiry learning apps, using Graasp [1], a social platform making use of OpenSocial. OpenSocial is an open specification, which defines a container and a set of common APIs for web-based applications. The App Composer will provide customization capabilities for OpenSocial apps, making it possible to customize such apps in different ways. For now two types of customization are provided, through two modules: a *Translator Composer* and an *Adaptor Composer*.

II. Background

A. Teacher development tools

The design of hybrid interactive learning environments for children and adults is a source of interest for instructional technologists [2]. School teachers and non-expert programmers make use of a selection of tools to create new learning apps or modify existing ones. In content authoring dynamic elements are mixed smoothly with static documents. Raptivity[2] or Zebrazapps[3] are examples. Support for mobile platforms has attracted attention [3].

Nowadays, most software editors generate a low level of interaction in the resulting media. Though UI design for

[1] http://www.go-lab-project.eu
[2] http://raptivity.com
[3] http://zebrazapps.com

learning must make people think and perform, UI design in general is about efficiency [4]. However, interactivity is fundamental in order to give students an active role and is a key feature provided by serious games, simulations and labs in e-learning scenarios to reinforce teaching of STEM (Science, Technology, Engineering, and Mathematics) principles [5].

Apps tend to be more interactive than standard Web pages, but currently the technologies are similar. XML and HTML make it possible to store the information, in such a way the final layout can be selected by the user and adjusted to the output device. Internationalization and localization are two other aspects of software that affect the whole creation process [6]. Also, many apps have been developed with Flash and Java. These became the defacto standard for interactive Web content due to the possibility of embedding them and the limitations of HTML. This is no longer necessarily the case [7]. Among the disadvantages of non-standard proprietary components such as these are severe security issues, the need to install them on each device, and the lack of mobile support. Applications based on the modern HTML5 stack are now appearing and generally preferred.

Another trend has been to merge dynamic assets with lessons that where previously static, through slideshows, wikis, Learning Management Systems and custom e-learning solutions. Courses provide and include SCORM packages, and try to make it easy to create and edit content and to not require server-side components. Sometimes they rely on specialized editors. In a recent work [8], an analysis of the existing web application toolkits is provided, and their main features, requirements and issues explored. It highlights the need to provide an authoring tool to let teachers build high-quality interactive applications which are customized to the student's cognitive progress within a particular course.

B. *OpenSocial as a framework*

OpenSocial[4] is an Application Programming Interface (API) for web-based social network platforms. Its primary goal is to provide a common framework that developers can use to ensure interoperability across various social networks on the Internet, which act as containers for each OpenSocial-compliant app. A particular app container must be chosen. Apache Shindig[5] is the most common and widely used. A set of learning tools such as Graasp [1], are based on Shindig to provide e-learning services. Through this framework social media platforms can be created for educational purposes [9]. The Go-lab portal, which contains a repository of online labs makes use of this stack, and the App Composer interacts with Shindig as well.

III. App Composer

The App Composer is an Open Source initiative[6] for developing a web application that enables teachers to translate and adapt apps which follow the OpenSocial standard. This is important in the context of Inquiry Learning Spaces because teachers can compose their own and include existing apps in them. The App Composer system is currently divided in two *composers* with different targets.

A. *Adaptor*

When trying to adopt apps and labs teachers often find that they don't really suit their teaching methods or the content does not match their course or the skills of their students. The Adaptor focuses on enabling teachers to derive their own apps by customizing "templates". Software developers write their apps as templates and provide two views:

- *Edit view:* For the teacher to customize the app. Contains parameterization options.
- *Watch view:* For the students to use the app. What the students will see depends on what the teacher configured in the Edit view.

The Adaptor is easy to use for both teachers and students. Teachers simply have to login into the AppComposer and choose the Template. The Template is simply an OpenSocial application which has been developed to support customization by including the specified views. Once they have selected the template, they are immediately shown the Edit view that belongs to that Template app, and they can use the specific functionalities for customization that the app offers. Once they are done customizing, they can publish it easily to students through the AppComposer itself. Students will simply have to follow a public link –which could also be added into an ILS– and immediately see the app in its customized form. That is, the Watch view, which will work according to the teachers' customization.

Complex Adaptors can be built through the use of a Python-based plug-in system which allows scripts to run both client and server-side code. A developer may provide an adaptor for mathematics where the teacher in the *edit* view selects a set of operators (e.g., sum, minus, log) and in the *watch* view students can take these operators and see generated graphics made on top of them with values they select. This way, the teacher could easily customize the mathematics template so students play only with some operators in each lesson.

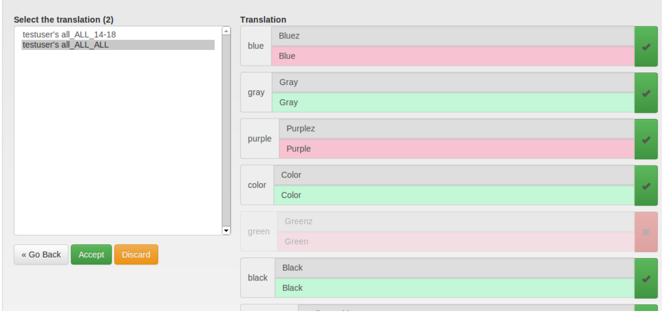
B. *Translator*

To reach schools around Europe, labs and apps need to be provided in the local languages. Students, especially young ones, often do not understand the original app's language. The original authors of the app can't be responsible for translating the app, because they do not necessarily know

[4] http://www.opensocial.org

[5] http://shindig.apache.org

[6] http://github.com/porduna/appcomposer/



Select the translation (2)

testuser's all_ALL_14-18
testuser's all_ALL_ALL

Translation

blue	Bluez Blue	✓
gray	Gray Gray	✓
purple	Purplez Purple	✓
color	Color Color	✓
green	Greenz Green	✗
black	Black Black	✓

Figure 1. Translator Composer's merge screen

which translations will be required, or have the resources and language skills required. The purpose of the *Translator Composer* is to take an existing *OpenSocial* app (to which they need to conform) and to let teachers translate it easily into other languages.

Languages and territories are supported. However, it is often the case that there exist significant communicative differences between geographical areas, depending on age or background. One of the goals of the Translator is to be able to translate apps for specific target groups as well. For instance, a specific "English for Young People" translation.

The workflow for teachers is the following:

- Register in the App Composer's website, which provides access and management features to all composers.
- Specify the location of the app to translate (the URL of the app's OpenSocial XML spec). The system automatically extracts translation and stores it in the server.
- Add translations.
- Publish the app through a public link.

Students do not need to register or log in, they can simply see the translated app through the provided URL. Teachers translating the same app are automatically aware of each others' translations and can merge them (see Figure 1).

IV. Conclusions and Future Work

An application composer for school teachers (the App Composer) has been described. Developed within the context of the Go-Lab Project, its target is to make it easy for teachers to compose apps easily from existing ones. The current version provides a set of particular composers: the *Adapt Composer* and the *Translate Composer*. The former enables teachers to take existing template apps and to customize them for their target domain. The later takes existing OpenSocial applications and enables teachers to translate them.

However, teachers with a more technical background often want to modify a particular application much more extensively, or even to create their own customized application. To satisfy the needs of this group, a possible *Builder Composer* is being evaluated. With a broad scope, it would permit the creation of a new OpenSocial app from an existing set of components and support some logic capabilities. The design could be loosely based on a tool such as *Easy Java Simulations* [10], albeit simpler.

Acknowledgment

This research was partially funded by the European Union in the context of the Go-Lab project (Grant Agreement no. 317601) under the Information and Communication Technologies (ICT) theme of the 7th Framework Programme for R&D (FP7. This document does not represent the opinion of the European Union, and the European Union is not responsible for any use that might be made of its content.

References

[1] E. Bogdanov, F. Limpens, N. Li, S. El Helou, C. Salzmann, and D. Gillet, "A social media platform in higher education," in *Global Engineering Education Conference (EDUCON), 2012 IEEE*. IEEE, 2012.

[2] L. P. Rieber, "Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games," *Educational technology research and development*, vol. 44, no. 2, pp. 43–58, 1996.

[3] L. SERAFIMOV, "Html5 support in mobile learning tools," in *Conference proceedings of" eLearning and Software for Education"*, no. 02, 2012, pp. 283–286.

[4] M. W. Allen, *Michael Allen's Online Learning Library: Successful e-Learning Interface: Making Learning Technology Polite, Effective, and Fun*. Wiley. com, 2011, vol. 3.

[5] H. Leemkuil, T. Jong, and S. Ootes, "Review of educational use of games and simulations." 2000.

[6] S. Auer, M. Weidl, J. Lehmann, A. J. Zaveri, and K.-S. Choi, "I18n of semantic web applications," in *The Semantic Web–ISWC 2010*. Springer, 2010, pp. 1–16.

[7] L. Rodriguez-Gil, P. Orduña, J. García-Zubia, I. Angulo, and D. López-de Ipiña, "Graphic technologies for virtual, remote and hybrid laboratories: Weblab-fpga hybrid lab," *2014 10th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, pp. 163–166, 2014.

[8] M. Latorre, A. Robles-Gómez, L. Rodríguez-Gil, P. Orduña, E. Sancristobal, A. C. Caminero, L. Tobarra, I. Lequerica, S. Ros, R. Hernández, M. Castro, D. Lopez-de Ipia, and J. Garcia-Zubia, "A review of webapp authoring tools for e-learning," in *Accepted to Global Engineering Education Conference (EDUCON), 2014 IEEE*. IEEE, 2014.

[9] E. Bogdanov, C. Ullrich, E. Isaksson, M. Palmér, and D. Gillet, "From LMS to PLE: A step forward through opensocial apps in moodle," in *The 11th International Conference on Web-based Learning (ICWL 2012)*, ser. Lecture Notes in Computer Science. Springer, 2012.

[10] F. Esquembre, "Easy java simulations: A software tool to create scientific simulations in java," *Computer Physics Communications*, vol. 156, no. 2, pp. 199–204, 2004.