Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-29 (2015) 129–133

An Architecture to use Easy Java-Javascript Simulations in New Devices [★]

Jacobo Saenz [*] **Francisco Esquembre** [**] **Felix J. Garcia** [***]
Luis de la Torre [*] **Sebastian Dormido** [*]

[*] *Computer Science and Automatics Department, Computer Science School, UNED, Juan del Rosal 16, Madrid, Spain(e-mail: jacobo.saenz@bec.uned.es, ldelatorre@dia.uned.es, sdormido@dia.uned.es).*
[**] *Mathematics Faculty, Universidad de Murcia, Campus de Espinardo, 30071 Murcia, Spain(e-mail: fem@um.es).*
[***] *Departament of Computer Engineering and Technology, Informatics Faculty, Universidad de Murcia, Campus de Espinardo, 30071 Murcia, Spain(e-mail: fgarcia@um.es).*

Abstract: Nowadays, virtual and remote laboratories play an important role in the students' learning process. For distance education in scientific areas, where the laboratories are an essential part, this importance is especially relevant. In the last years, the newly discovered Java vulnerabilities and the new devices which do not support Java applications (tablets and smart-phones), make even harder to run or develop these laboratories as Java applications. On the one hand, a non-digitally signed applet cannot be run in a web browser or in a smart-phone. On the other hand, portable devices, like a pc-tablet, have limited computation resources, and sometimes it is not enough to run complex simulations. EjsS (Easy Java-Javascript Simulations) is an open source tool used for creating all kind of simulations by introducing the equations of the model, and building a graphical user interface (GUI) for the application. This work presents a solution to the two previous problems that appear when using EjsS. Both are resolved using a Java model that runs in a server and takes the heavier computational load. This Java model is linked with a Javascript GUI in the client device. The link is based on a web-socket connection between server and client using a JSON format. The proposed solution has additional advantages, such as the possibility of multiple users working with the same model, or reusing an already existing Java application by just building a new Javascript view.

Keywords: Education, Laboratory, Control, Simulation, Digital Computer Applications.

1. INTRODUCTION

Nowadays online resources have become widespread, and they have an important role in the learning process of students for all kind of studies. Thanks to new technologies and devices, the students can complement their lessons with a wide range of applications, videos, simulations or additional information. In this regard, for scientific areas, applications that simulate physical systems are receiving more and more attention. These simulations generally have an intuitive graphical user interface (GUI) that supports some level of user interactions and control over the system.

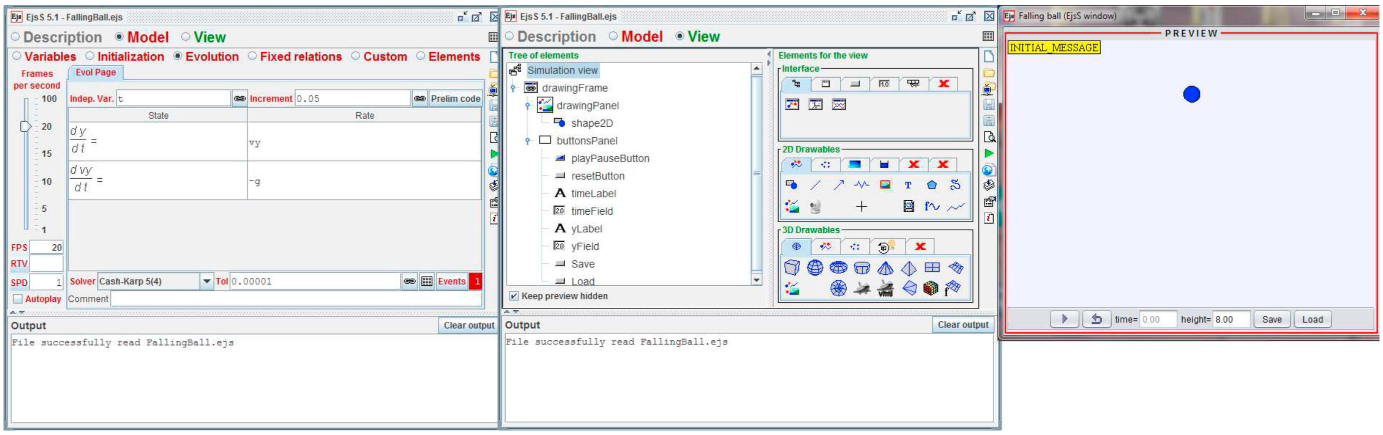
In the last years many universities have added this kind of tools to complement their traditional courses: Guimaraes et al. (2011); Vavougios and Karakasidis (2008); Restivo et al. (2009); Garcia-Zubia et al. (2009); Farias et al. (2010); Yuliang Qiao and Hu (2010); Andreja Rojko (2010); Yazidi et al. (2011); Hassan et al. (2013); Santana et al. (2013); Tawfik et al. (2013); Bose (2013)). Considering the distance education paradigm, in scientific and technical areas, a face-to-face laboratory practice is not commonly available for students. For that reason, the use of online applications, such as virtual laboratories (VLs), remote laboratories (RLs) or both (VRLs), is essential in the distance learning process.

However, creating a new virtual or remote laboratory is not an easy task. The software must usually be self-made and, most of the time, this task has to be performed by teachers, researchers or students. Normally, these applications are created using integrated development environments (IDEs) of high level programming languages or with tools for creating animations. Currently, most of these tools, to a greater o lesser extent, use Java. Unfortunately, over the recent years many Java vulnerabilities have been appearing. Therefore, restrictions for running digitally signed Java applets make very difficult the dissemination and use of VRLs which are created based on this technology. In addition, new mobile devices with Internet access, like smart-phones or pc-tablets, do not support Java. This constitutes another impediment for the publication of VRLs that rely on this technology.

To address the unsupported Java problem some developers have written their VRLs using Javascript, such as, for example the works by Frank and Kapila (2014) and Glotov et al. (2013 (2nd). This solution solves the two problems mentioned above but it involves a new one due to the limited computational resources of a smart-phone or tablet pc, which sometimes it is

[★] Sponsor and financial support acknowledgment goes here. Paper titles should be written in uppercase and lowercase letters, not all uppercase.

2405-8963 © 2015, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2015.11.225



$\frac{dy}{dt} = $ vy

$\frac{dvy}{dt} = $ -g

```
File successfully read FallingBall.ejs
```

Fig. 1. Main view of the EjsS editor in Java enabled version

not enough to run complex simulations. In addition, rebuilding an already existing VRL from scratch in Javascript can be a huge task.

The present work applies a solution for all these previous problems when using Easy Java/Javascript Simulations (EjsS) for creating the VRLs applications.

The paper is organized as follows. Section II describes the actual state of EjsS and how it works for creating both, Java and Javascript applications. Section III discusses the architecture proposed in this work to expand EjsS in order to give answer to some actual problems with Java and Javascript applications. Finally, Section IV gives some final conclusions and describes further work.

2. EASY JAVA/JAVASCRIPT SIMULATIONS

2.1 The EjsS Tool

EjsS is an open source authoring tool designed for students and teachers who need to make fully functional applications and simulations in scientific and engineering areas. This tool offers an easy way to create a GUI for simulations, according to the user needs of interactivity and visualization. EjsS allows the user to create applications in both Java or Javascript. This applications that can be standalone (in the case of Java) or that might be run in a web browser (Java and Javascript).

The main strength of EjsS is that it eases the development of applications by teachers and students who want to focus in the simulation itself and not in the technical programming aspects Farias et al. (2010); Chacon et al. (2015).

When an application built in EjsS is finished, the user can run it using the EjsS editor. Then, the application opens in a new window ready-to-use. Another option is to package the application in order to run it later as in standalone mode (for Java applications) or inside a web page (for Java and Javascript ones).

2.2 The EjsS Java mode

EjsS applications are structured in two main parts, the view and the model:

- The *model* can be seen as Java programming code, differential equations (left image in Figure 1) and/or connections to other software or hardware. The simplicity or complexity of the model depends only on the users requirements and their knowledge of the system they want to simulate.
- The *view* provides final users a GUI, whose elements have been added one-by-one, and determines the interaction and visualization capabilities of the application. This view is built with a tree architecture by dragging and dropping elements in the right panel of the EjsS editor (middle image in Figure 1). A previsualization of the GUI built in this way is also shown by EjsS (right image in Figure 1).

2.3 The EjsS Javascript mode

The problems with Java vulnerabilities was solved by EjsS in a previous release (5.0) by using the Javascript programming language instead of Java. Therefore, with EjsS 5.0 or above, users can develop new VRLs based in Javascript with a little Javascript knowledge.

When running this mode, the main structure of EjsS does not change in the eyes of the user, and building of an application is very similar to the previous Java-based case, as Figure 2 shows. Actually, at a first sight, the only difference between is that the name of the tab changes from *View* to *HtmlView*.

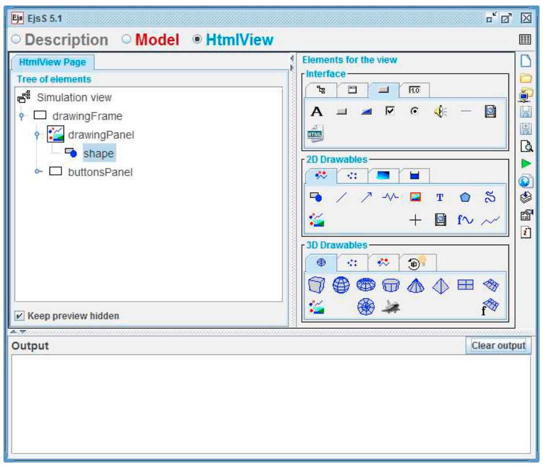


Fig. 2. Main view of the EjsS editor, in Javascript enabled version