

eXULiS – a Rich Internet Application (RIA) framework used for eLearning and eTesting

R. Jadoul^{*,1}, P. Plichart¹, J. Swietlik¹, and T. Latour¹

¹ Research Centre Henri Tudor, 29 John-Fitzgerald Kennedy, L-1855 Kirchberg, Luxembourg

Today building Rich Internet Application (RIAs) still requires programmer's skills. The rare open frameworks which allow non-technical users to design rich content education modules are quite limited in terms of presentation and certainly hinder the creativity.

EXULIS is a free, open-source cross-platform application that is able to display rich interactive graphical interfaces in Web browsers. Its power resides in the extensible capabilities of a versatile reader that accepts standard graphical formats like XUL and SVG, and that produces the expected behavior. The underlying framework of EXULIS is based on a 3 years experience in the domain of distributed Internet-based eLearning and eTesting experiments. This framework has been designed to be extensible, and thus it could be enriched with new thematic layers (mathematics formalism, chemical descriptions, and so on) related to some markup languages (e.g. MathML, ChartML, ChemML). These mark-up languages will support the creation of some types of learning and testing units.

This paper explains the basics of the EXULIS architecture as well as the extensibility mechanisms and it will also present a concrete use case of this program.

Keywords EXULIS; RIA; XUL; SVG

1. Introduction

Since the Internet has been presented as a revolutionary medium, a lot of crucial issues are still to be addressed. Its richness is certainly not to be proved as long as one only considers the contents. The number of the libraries now offering an online version of their catalogues is increasing. Some of them even propose digitalized documents covering a growing part of their book collections. Universities are also publishing their syllabi, lectures, and other materials via the Internet. Contents may even be directly elaborated on the Web itself via online newspapers, free encyclopaedias based on Wiki technology, etc.

In this sketch, everything seems to be right. But, are the online contents *ergonomically* accessible? Does any person (e.g. an older person, a 6 years old child, a Ph.D. candidate or a blind person) who is in front of an Internet connected device have an easy and fruitful access to the published materials? With these questions in mind, we understand that there are some limits of this revolutionary medium.

Rich Internet Applications (RIAs) seem to be the solution offered by the computer-scientists for the more and more demanding users. Unlike the previous generation of Web Applications, RIAs plan the leverage of the user experience with more powerful graphical user interfaces (GUI) including charts, drag & drop, etc. RIAs also aim to remain as platform-independent and setup-free as possible. One should be able to work on data and tools both available online and these tools must be usable without setup or configuration on the client computer. However, RIAs also should be more than a few good-looking GUI because the desktop applications have made the users accustomed to a certain level of responsiveness and customizability.

The RIA solutions are now legion. These are proposed to the developers as frameworks. Nearly none of these frameworks is really standardized even if the majority rely on the XML as a base for at least the GUI mark-up. The well-known application/UI formats are Mozilla's XUL, Microsoft's XAML, Adobe's MXML, Laszlo Systems' LZXML, ActionStep's ASXML, ASWing's AWML and enFlash's ML.

* Corresponding author: e-mail: raynald.jadoul@tudor.lu, Phone: +352 42 59 91 242

In this paper, we describe a framework named eXULiS. It is not a standard. It is not the ultimate RIA solution but rather another alternative that has been matured for the last three years and that tries to stay as close as possible with the standards worked out for the moment. Thus, it is a work in constant progress. This document first gives an overview of the platform and the place of eXULiS in the whole schema. Afterwards the architecture of the eXULiS' framework will be explored. Then, the extensibility mechanism will be explained and a case of use will be briefly presented.

2. The overall architecture of the TAO platform

The Research Centre Henri Tudor started to develop a CBA platform called TAO (in French “*Testing Assisté par Ordinateur*”) three years ago. The foundations of this platform rely on two main components. The first part is located on the server side and this part is responsible for structuring, storing, and sharing of the data. It is called Generis⁴. The second part is active on the client side and it is in charge of the correct display and completion of the tests and the questions (or items). It relies on the Flash player plugin that is installed on a large computer base (amongst individual computers).

The work of this client-side component is to receive and to interpret the files describing the tests. The syntax of a test description is a mix of XUL and QML. XUL (or “XML User Interface Language”) gives the layout of the graphical objects to appear on the screen of the computer for the test. We choose XUL for its maturity and because it was more open and community oriented than some other initiatives. QML stands for “Questions Markup Language.” The QML used in the TAO platform was extended to encompass the needs of the IRT model as well as the new requirements towards the multilingual assessment.

The heart of the rendering engine was a parser called XUL2SWF (where SWF is the file extension of the Flash movies). The new framework eXULiS is more than a simple evolution of the XUL2SWF engine. It now contains a XUL parser completely rewritten to be extensible and more compliant with the Mozilla specifications. It also includes a second parser that is able to display SVG drawings. SVG is the acronym for “Scalable Vector Graphics.” This XML language is used for describing two-dimensional vector graphics. The integration of a drawing format was required to open new vistas for the design of RIAs and for the authoring of advanced types of tests and items on the TAO platform.

The authors who create new graphical layouts for RIA and/or for CBA can proceed using the tools freely available on the Web; XUL files can be written with *xuledit.xul* and SVG files can easily be produced with *InkScape*, for example. Our team of the Research Centre Henri Tudor is also developing an authoring tool that will produce the XUL and SVG layout files. This tool will complete the TAO platform.

In the Figure 1 “TAO platform architecture,” you can discover the interactions of the two components described here above in order to deliver the RIAs, and more specifically, the tests to the tested subjects. This schema depicts at the same time the topography (or deployment architecture) and some sequences of actions between the units involved in the TAO platform. This schema should be read from left to right as everything in our platform starts from Generis [1].

On the server side, Generis provides, via its PHP API or via its Web Services API, two sets of information issued from the RDF triples it manages. The first of these sets of data can be used by 3rd party applications (Authoring tools, eLearning platform, eBusiness applications, and so on) to produce the RIA description files (XUL, SVG, Cascading Style Sheets and JavaScript) that can be either natively rendered by the browsers embarking the Gecko Runtime Engine technology or that can be displayed in all the families of browsers via the eXULiS plugin. The second set of data contains the same kind of files as the ones involved in the first set but it also includes files delivered in a format that is not directly interpreted by the Internet browsers (even the Gecko family). This part of the second data set is *solely* dedicated to the eTesting. It holds some definitions that only have a meaning in the context of a CBA.

The files formats specifically used by eXULiS for CBA are: the TAO QML definitions, the XPD (XML Process Definition Language) and some specific XML and RDF datasets. Please note that the RDF information are also nested in XUL and SVG description files for two purposes: the WAI-ARIA – Web Accessibility Initiative / Accessible Rich Internet Applications – effort and to allow the TAO authoring tool to reload efficiently the Test/Items definitions.

The TAO QML files contain the logic and the hierarchical structure of the assessments. It means that the files describe a specific assessment in terms of a campaign involving one or several sequences of tests (potentially available in different languages) including one or more sequences of Items made of a set of particular Items, each one integrating a Problem (stimulus), and Inquiries composed of a Question and a Distracter (e.g. a set of Proposals for multiple choices question, an open text, a puzzle, and so on).

In the case of predefined sequences (called *scenarii*) of Tests, Items or Inquiries, eXULiS evaluates, at a moment T, the execution context of the assessment; then it uses some Workflow (WF) definition files formatted in XPDL that contains the conditions of the time T, to display the correct user interface at time T+1. As mentioned above, the definition of the GUI is not stored in XPDL but in XUL and SVG files.

Generis⁴ + eXULiS architecture for TAO

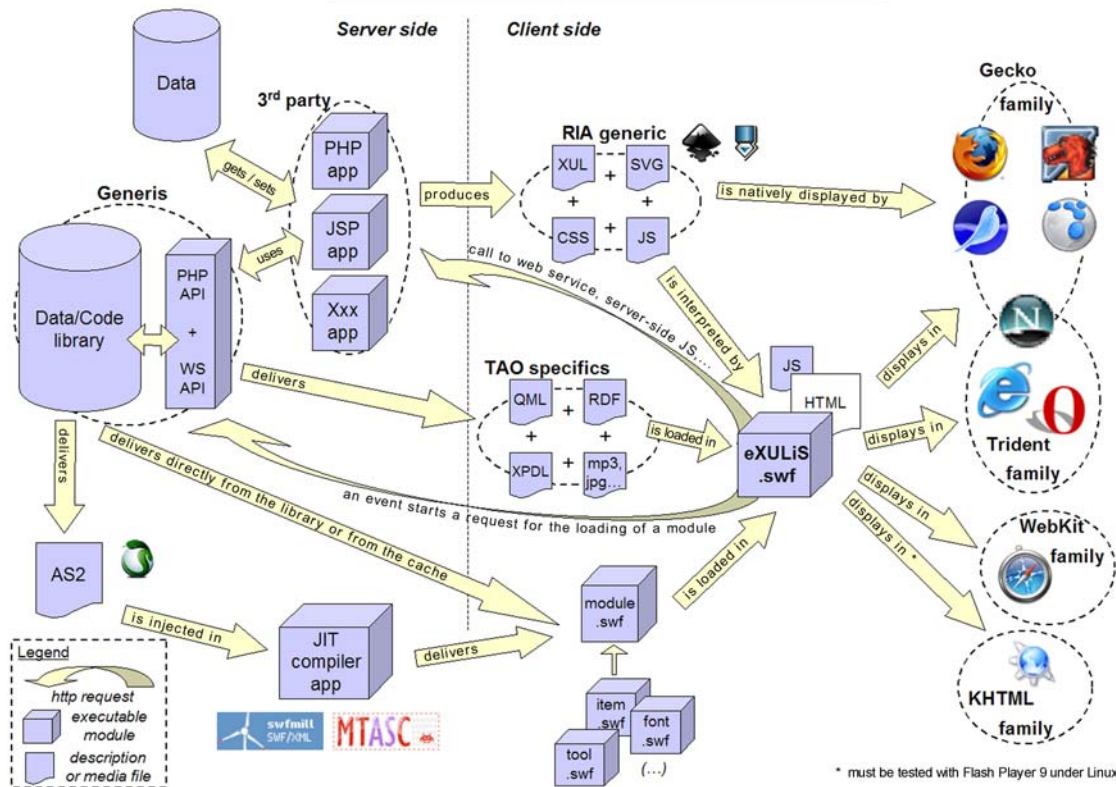


Fig. 1 TAO platform architecture. This deployment view shows the place of eXULiS in the architecture.

Another interesting aspect of eXULiS is its capacity to extend its dynamic behaviour in numerous manners. The engine is a Flash movie (.swf file) and it can act as a relay to local or remote function calls. It embarks a set of wrappers and API that enables the invocation of Web Services, server-side JavaScript, local JavaScript (located in the Web page – HTML, PHP, JSP – that nests the eXULiS Flash object), remote CGI scripts, and it even allows the communication with client-side desktop applications through the use of local connections.

When it detects a specific need (for example, a Test event is raised requesting that the current Item is displayed in Japanese), eXULiS may forward this event to the Generis back office that will provide (if necessary via a Just In Time compilation) the useful resources (in this case a .swf module containing some Hiragana, Katakana and Kanji character sets). The .swf modules may contain diverse types of resources including fonts, tools (e.g. calculator, notepad), compressed XML datasets, media, and so on.

In the next sections, we will briefly discuss the eXULiS and explain how its modular internal architecture is a favourable ground to extend its capabilities.

3. Design of the eXULiS framework

3.1 A two-folds construction

The effort to build the eXULiS framework started before the diverse initiatives led by the standardization agencies (e.g., W3C). Instead of creating a homemade GUI format for the specific needs of the CBA, we decided to select one that was already available. We wanted an open standard, well established with enough resources available online, and if possible, a solid community has supported it; the Mozilla project called XUL was our great preference even if this XML-based description format was not a standard.

For our needs, we implemented this format in a component that we named XUL2SWF. Basically, this was a monolithic class encapsulating a parser that was recursively interpreting the XUL tags to render the corresponding graphical user interface using the Adobe (formerly Macromedia) graphical widgets (e.g. basic *movieclips* and *v.2 components*). At that time, about 60% of the XUL widgets were implemented in the XUL2SWF engine. This was enough for most of our needs.

Today, with eXULiS, the one-piece class has become a real framework and the coverage of the XUL specification increases as well as the new engine embarks other XUL affiliated technologies like XBL (eXtensible Bindings Language) and RDF (Resource Description Framework).

Furthermore, to fulfil the new requirements elicited during three years of use of the TAO platform, a few months ago, the development team decided to add to the XUL framework, a second framework to handle the SVG standard.

In the Figure 2 “eXULiS framework overview,” the two parts of the class tree can be clearly identified: on the left side, the XUL classes are inheriting from a common *XULelement*, and on the right side, the SVG classes are inheriting from a common *SVGelement*; both *XULelement* and *SVGelement* are inheriting from *element*. The ancestor class *element* acts like a relay allowing a natural communication between the two frameworks, in particular via the events. For example, a widget in the SVG framework can subscribe to any events of the type *xyz* and start to listen while a widget in the XUL framework can dispatch this type of events. The *xyz* event *bubbles up* to the root of the frameworks and it can cross this bridge to be broadcasted to the subscribing SVG widget.

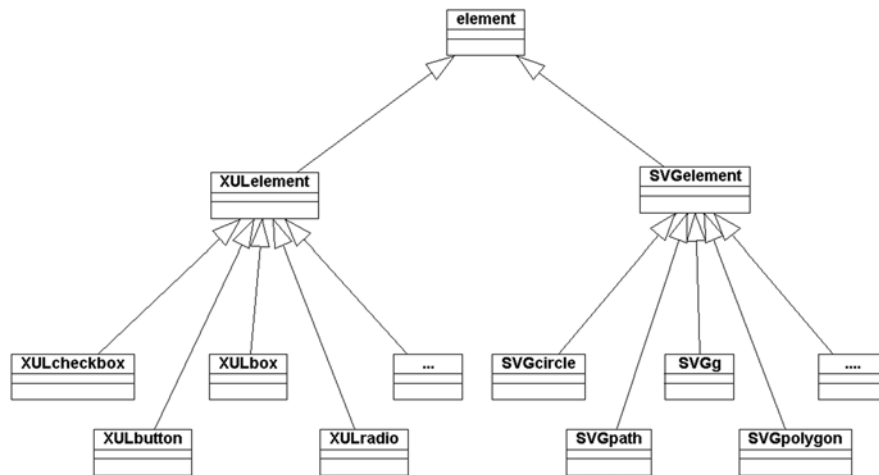


Fig. 2 *eXULiS framework overview.* This class diagram shows the two-folds aspect of the framework.

The whole framework can also use some other static classes available in the Flash framework; an obvious example is the *Tween* class that allows some movements and transitions in the state of the widgets. The development team will probably, in a future version, integrate this class as an element of the SMIL (Synchronized Multimedia Integration Language).

3.2 How can eXULiS be extended?

The integration of the SVG standard in eXULiS first intended to allow the authors to create their own custom themes and skins for the tests/items (e.g. a button with shape of a cloud for 6 years old children). The power of the SVG standard and the capacity of Flash to call some external JavaScript functions, unleashed the potentials of eXULiS. First, we created a module transforming some ChartML tags into SVG (see Figure 3) that eXULiS displays perfectly. Our latest project targets some needs in physics.

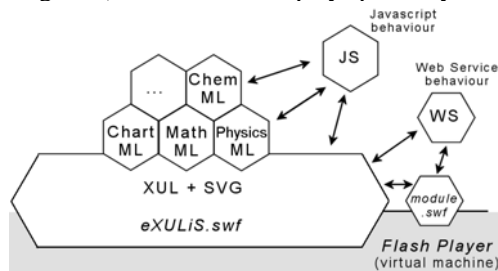


Fig. 3 The eXULiS extension. This diagram shows how new markup languages may be rendered.

3.3 An example of use of eXULiS

This example (on Figure 4) is a mix of physics rules (gravity, levers, axis) applied to a schema that is a composition of SVG drawings and XUL widgets (buttons, checkbox) interacting in a laboratory allowing to experience a problem of static physics science.

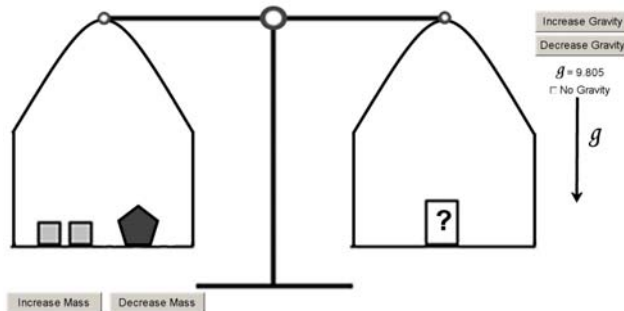


Fig. 4 A physics lab with eXULiS. The figure shows a physics laboratory using SVG and XUL widgets.

4. Conclusion

With eXULiS, we expect to reach a more advanced ergonomic level in the GUI of the RIAs and CBA and we are happy to find out that the most recent works of the W3C go into the same direction we already chose some time ago. Nevertheless, we continue to consider this engine as a work in progress.

For the enhancements of eXULiS, the developers of the TAO platform pay a particular attention to the efforts done by the WAI group (the Web Accessibility Initiative aims to make the Web accessible to people with disabilities) and we especially keep informed about the works lead by the ARIA (Accessible Rich Internet Applications) working group [2].

Acknowledgements The first author really thanks Ms. Sachie Mizohata for her indefectible support.

References

- [1] P. Plichart, R. Jadoul, T. Latour and L. Vandenabeele, Communication at the Advances in Intelligent Systems – Theory and Application AISTA2004, Luxembourg (2004).
- [2] R. Schwerdtfeger, Roadmap for Accessible Rich Internet Applications, W3C Working Draft 26 Sept. 2006