INTERACTIVE WEB-BASED ANALYSIS CLIENTS USING AJAX: WITH EXAMPLES FOR CMS, ROOT AND GEANT4

G. Aleverson, G. Eulisse, S. Muzaffar, I. Osborne, L. Taylor, L. A. Tuura,

Northeastern University, Boston, U.S.A.

Abstract

We describe how a new paradigm dubbed AJAX (Asynchronous Javascript and XML) has enabled us to develop highly-performant web-based graphics applications.
Specific examples are shown of our web clients for: CMS Event Display (real-time Cosmic Challenge), remote detector monitoring with ROOT displays, and performant 3D displays of GEANT4 descriptions of LHC detectors. The Web-client performance can be comparable to a local application. Moreover the web client does not suffer from any of problems of software packaging, distribution and installation and configuration for multiple platforms. AJAX, which uses a mixture of Javascript, XML and DHTML, is spreading rapidly, helped by its support from industry leaders such as Google, Yahoo and Amazon. We describe how AJAX improves on the traditional post/reload web-page mechanisms, by supporting individual updates of sub-components of web pages and explain how we exploited these design patterns to develop real web-based High Energy Physics applications.

INTRODUCTION

The standard web page workflow

The adoption of web interfaces as application for thin clients has always been difficult because of the intrinsic lack of interactivity and responsiveness of the regular HTTP workflow.

Such a workflow can in-fact be characterized in 4 steps:

- HTTP GET request
- Server side creation of the full page
- Transmission of the full page
- Client side HTML parsing and page display.

Until this workflow is completed the user cannot interact with the application and most important, the full complete web page has to be reloaded when the user asks for more information, even if the differences between the two pages are minimal (think for example about the expansion of one of the branches of a tree browser). It is therefore clear that this kind of workflow, albeit very simple to implement, is too simple and too bloated for creating interactive applications.

The AJAX workflow

To overcome these problems a new technique has found its way in the IT industry, exploiting the capabilities of a special Javascript object called XMLHTTPRequest.

This technique, dubbed AJAX, allows to overcome the high latencies of the standard workflow for web pages and to obtain highly interactive web applications that behave exactly like desktop ones, but with the advantage of being web based, hence platform independent and easily deployable to users.

The initial workflow of AJAX web applications is similar to the one of standard web pages: a request is done to the server which replies with some sort of HTML. The difference is that this time the server returns the bare minimum information: a web page contains a Javascript script that does the second part of the workflow.

- Javascript triggers an HTTP GET request using XMLHTTPRequest object.
- The servers elaborates the request and prepares an xml response describing the changes between the old page and the new one.
- This delta information is sent back to the client
- Which elaborates the response in an asynchronous javascript event and prepares the HTML fragments to update the page from the XML received by the server.
- The page DOM is finally updated in by the javascript script.

This workflow is then iterated whenever the user requires more or updated information or if the clients feels like it has to refresh some information which could be outdated.

The AJAX advantage

While the initial integrated data transferred by an AJAX web application is similar to the one of a traditional web page (but loaded in smaller chunks), it's on subsequent page reloads that AJAX shows its strength.

A simple tree expansion or update of only one part of the display is treated as such, without the need of reloading the rest or separating elements in different FRAME elements. By breaking the GET/PROCESS/RELOAD paradigm of traditional web pages all AJAX allows the server and the client to communicate without reloading the page, hence allowing them to always work on deltas of information, saving processing time and bandwidth.

IGUANA & AJAX

IGUANA OVERVIEW

IGUANA stands for Interactive Graphics for User ANAlysis. It is a C++ toolkit and framework which is particularly designed to provide graphics rich, interactive, applications to the physicists. It is already providing different levels of event displays and visualization applications for CMS experiment users. Up to now the IGUANA architecture consisted in four different layers a

- A set of adaptors to various data sources
- An object model for the representable elements found in data sources
- A set of representation technology for the object model representable (3D, 2D, Lego, ROOT histograms, text, XML)
- A modular Qt based application framework with different plugins providing custom analysis application.

This modular approach to the creation of applications has the great advantage that the actual QT GUI is only one of the many components and it is only loosely coupled to the rest of the application. This eases the task of removing such a component and providing another kind GUI.

AJAX layer for IGUANA

On top of this already proven and reliable framework we have added a few components to facilitate the writing of AJAX enabled versions of those applications. One of the goals of this addition of extra components is to reuse as much as possible what was already written for the QT version.

The new components consists in:

- a simple, embedded HTTP/1.1 server
- a web service framework sitting on to of it
- a javascript library for creating AJAX enabled web pages.
- a set of web pages written using the above library
- a set of web services to enable these pages to access the content provided by iguana.

The embedded HTTP/1.1 server and the associated web services are the bridge connecting the standard IGUANA applications written in C++ and interacting with the experiments software and the javascript web application. The design of the web server is as simple as possible, without too many options, since in practice it will be working hidden behind a real Apache webserver.

EXAMPLES OF AJAX APPLICATIONS USING IGUANA AJAX FRAMEWORK

Rationale behind the various examples

In order to demonstrate the power of the AJAX approach for developing applications we decided to create a set of example prototypes (now at different level of completeness) touching different areas that are crucial for an interactive application used by phisicists. In particular we selected the areas of interactive 3D graphics, of experiment framework interaction and root visualization.

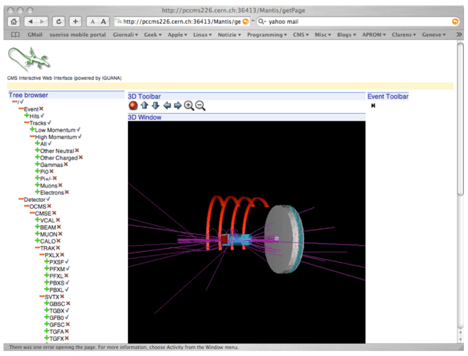
Geant4 Visualization

IGUANA provides a set of classes for looking at GEANT4 geometries in 3D. The desktop application uses Open Inventor toolkit taking advantage of the 3D acceleration of modern graphics cards.

The user can rotate the model in realtime just by dragging the mouse on the 3D window.

One of the biggest advantage of AJAX is that it enables to transport on the web interactive graphics-rich applications like in this case. A web service for generating 3D images out of the standard IGUANA objects (which in this case acts as proxies to the G4 geometry) was written.

This web service was then coupled to a custom Javascript widget which acts as a 3D browser and that reacts to mouse dragging events by requesting to the server a new, updated, image. The server elaborates the request by creating the requested image, fully exploiting its hardware acceleration features for 3D, and sends the image back to the client, which then updates the outdated picture. On standard networks, including home ADSL to



There was one error opening the page. For more information, choose Activity from the Window menu.

Figure 1: IGUANA web application interacting with the old CMS simulation software.

CERN connections, this has been proven to be fast enough for realtime behavior. This approach has also the non-negligible advantage of being completely independent of the installation of accelerated 3D drivers on the client machine which is still a non resolved issue for Linux distribution (and in particular Scientific Linux 3) especially when running on a notebook.

CMS Visualization

Similar to the previous application, but somewhat more sophisticated, is the AJAX version of the CMS Visualization application.

In this case the problem is more complicated than with the Geant4 Visualization because, besides of the hurdles imposed by realtime 3D graphics also we have to interact with generally more complex framework of an experiment.

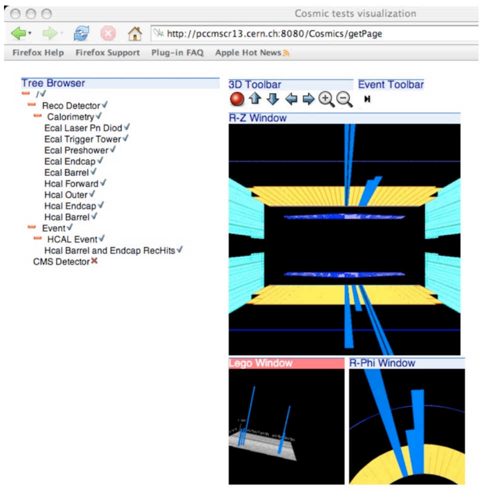


Figure 2: IGUANA web application interacting with CMSSW software framework reading real data coming from HCAL.

If we look at figure 2 it is also evident the ability of the Javascript GUI library to provide a fully context sensitive, multi-window environment which is found in the desktop version as well.

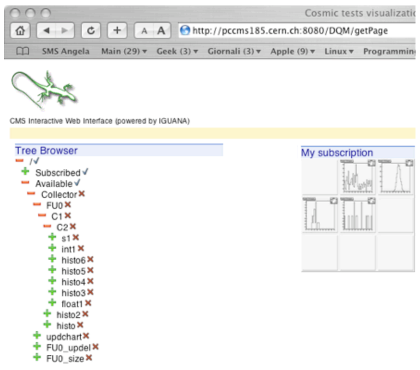


Figure 3: IGUANA web application interacting with CMS DQM framework and displaying results using a ROOT plotted histogram.

ROOT Visualization

Another field in which AJAX based web applications could provide a superior experience is in visualization of histograms. IGUANA already provides the ability of embedding ROOT histograms within its QT based GUI.

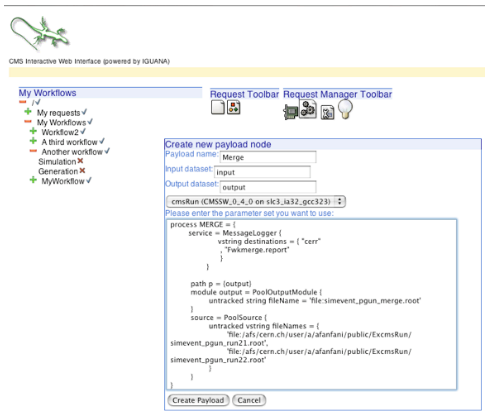
This has been for example used to develop a graphical user interface for the Data Quality Manager framework of the CMS experiment. Thanks to the very modular design of IGUANA, where the abstract objects description is separated from their actual representation, it was a matter of a few days adapt this GUI to run as a AJAX application running on top of the already present components. The AJAX advantage in this case is that we can update the ROOT window which has been updated on server side without the need of reloading the full page.

CMS Montecarlo Request System

The usage of the IGUANA framework for web applications is not limited to fancy, 3D rich, graphical applications, but can also be used for a more canonical web application like managing the requests of a Montecarlo production system.

The Javascript GUI library comes very handy to build web pages that, while resembling a more traditional layout, feature a set on characteristics, popup menus, dialogs, wizards, toolbars, common to desktop applications.

Moreover, since there was the need to interact with a traditional database we have developed a C++ web service for IGUANA that by taking advantage of the new toolkit for abstraction of database operation, CORAL (COmmon Relational Abstraction Layer), which allows



Payload name: Merge
Input dataset: input
Output dataset: output
cmsRun (CMSSW_0_4_0 on slc3_ia32_gcc323)
Please enter the parameter set you want to use:

```
process MERGE = {
        service = MessageLogger {
                vstring destinations = { "cerr"
                 , "Fwkmerge.report"
                 }
             }

        path p = {output}
        module output = PoolOutputModule {
               untracked string fileName = 'file:simevent_pgun_merge.root'
        }
        source = PoolSource {
               untracked vstring fileNames = {
                       'file:/afs/cern.ch/user/a/afanfani/public/ExcmsRun/
simevent_pgun_run21.root',
                       'file:/afs/cern.ch/user/a/afanfani/public/ExcmsRun/
simevent_pgun_run22.root'
               }
        }
}
```

Figure 4: a more traditional web application, the MCREQUEST request system prototype for CMS.

the interaction of our web application with a variety of database backends (Oracle, MySQL, sqlite).

FUTURE PLANS

Now that we have demonstrated a full blown desktop application can be reproduced on the web using AJAX, we would like to refine both the Javascript widget library to allow more complex and polished GUIs bringing it on par with the traditional desktop one, and the server, so that it can handle the load of the users of an experiment in data taking mode.

REFERENCES

[1] http://www.tifr.res.in/~chep06/.
[2] http://iguana.cern.ch/
[3] http://iguanacms.web.cern.ch/
[3] http://pool.cern.ch/coral
[4] http://www.apache.org/
[5] G. Alverson, G. Eulisse, S. Muzaffar, I. Osborne, L.A. Tuura, L. Taylor "IGUANA Architecture, Framework and Toolkit for Interactive Graphics", CHEP'03 proceedings, La Jolla, March 2003.
[6] G. Alverson, G. Eulisse, S. Muzaffar, I. Osborne, L.A. Tuura, L. Taylor "IGUANA: a high-performance 2D and 3D visualisation system", Nuclear Instruments and Methods in Physics Research Section A, v. 534, iss. 1-2 [SPECIAL ISSUE], p. 143-146.
[7] J. Apostolakis et Al. "Geant4 - a simulation toolkit ", Nuclear Instruments and Methods in Physics Research Section A, 2003.