RAPID APPLICATION DEVELOPMENT USING WEB-2.0* TECHNOLOGIES

S. Reisdorf, B. Conrad, D. Potter, M. Hutton, P. Reisdorf, LLNL, Livermore, CA 94550, USA

Abstract

The National Ignition Facility (NIF) strives to deliver reliable, cost effective applications that can easily adapt to the changing business needs of the organization. We use HTML5, RESTful web services, AJAX, jQuery, and JSF 2.0 to meet these goals. WebGL and HTML5 Canvas technologies are being used to provide 3D and 2D data visualization applications. jQuery's rich set of widgets along with technologies such as High Charts and Data tables allow for creating interactive charts, graphs, and tables. RESTful Web Services have replaced the traditional SOAP model allowing us to easily create and test web services. Additionally, new software based on Node.js and WebSocket technology is currently being developed which will augment the capabilities of our existing applications to provide a level of interaction with our users that was previously unfeasible. These Web 2.0-era technologies have allowed NIF to build more robust and responsive applications. Their benefits and details on their use will be discussed.

INTRODUCTION

It is often said that with software "the only thing constant is change" [1]. Software projects usually have a high degree of ambiguity where requirements are frequently changing and the scope of work is continually shifting. This volatility can make software development difficult and without the correct approach projects can either take too long or end up delivering the wrong product. Additionally choosing the right technology can be a challenging task, especially in a time when web technologies and platforms are continually evolving. In order to meet these demands, the National Ignition Facility (NIF) has adopted many web 2.0 technologies such as jQuery, Oracle Application Express (APEX), HTML5, Node.js, and Representational state transfer (RESTful) web services. Using these technologies NIF has been able to deliver many data-driven applications that are reliable, cost effective and can easily adapt to the changing business needs of the organization. This rapid application development approach has resulted in faster delivery times, better quality, lower cost, lower maintenance, and greater customer satisfaction.

SOLUTIONS

The evolution of Web 2.0 and HTML5 has created many frameworks that help make software development easier. Access to so many different tools is a blessing and a curse as frameworks and languages that seem like a good choice today might not be the best option tomorrow. Choosing the right tool or technology and knowing when to change is essential for an organization to succeed.

A few years ago the NIF invested in the Java Server Faces (JSF) framework. Recently we have found the JSF framework too rigid. JSF is a component based framework that abstracts many layers such as JavaScript, CSS and AJAX requests. This abstraction can be nice as developers don't need to be CSS or JavaScript experts; however, it is very challenging if you need to do something that the component or framework doesn't support. Recognizing this, we have switched to using more flexible frameworks such as Oracle APEX and jQuery.

JavaScript and jQuery

As the web has evolved dynamic and responsive applications are easier to create thanks to JavaScript and leading JavaScript libraries such as jQuery. jQuery is the industry standard JavaScript library [2] that abstracts many core JavaScript functions into a standard device agnostic API. It has an extensible plugin architecture allowing developers to create reusable "widgets" that can be plugged into any web application. This allows developers to quickly develop feature-rich user interfaces; however, care needs to be taken in order to prevent the maintenance issues associated with possible unstructured and unwieldy code. Unlike Java which has a compiler and IDE that can check and find most bugs during development, JavaScript is much more dynamic and is compiled in the browser at runtime, making it harder to debug. Adopting Object Oriented (OO) coding practices in JavaScript is essential to create testable objects that can easily interrelate with one another. A variety of open source frameworks such as jQuery-ui widget factory, JavaScriptMVC, Backbone.js, and AngluarJS provide this capability. Each framework has different features, some of which are lightweight while others are much more complicated. The NIF decided to use a hybrid approach using just the generic core objects from JavaScriptMVC along with the jQuery-ui widget factory to create an OO framework. This process allows us to be much more agile and adaptable as we are not bound to the constructs of any specific library. Leveraging OO design practices, NIF software developers have built up a library of components and jQuery widgets that are reused across applications. One such application that uses these widgets is the NIF Archive Viewer (Fig. 1). This application is a data driven portal into all experiment and diagnostic data.

*This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. #LLNL-ABS-632634, LLNL-CONF-644468

ISBN 978-3-95450-139-7

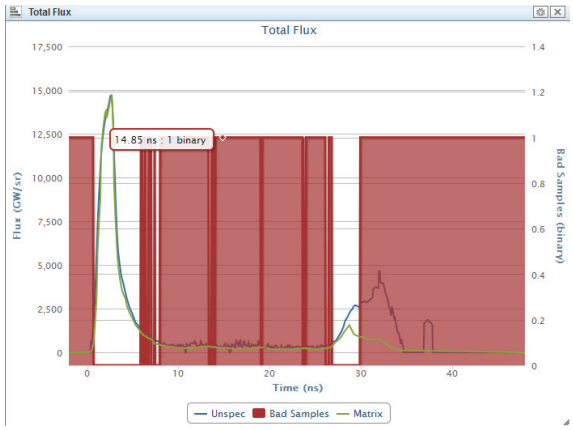

Figure 1: A configurable and reusable jQuery widget that creates responsive charts with hover, zoom, and click capabilities

Users are presented with a dynamic dashboard that renders tables, charts, graphs, and interactive discussions for the NIF experiments. The NIF uses the open-source JavaScript library HighCharts to create feature-rich charts and graphs that allow users to zoom, highlight and click into details on a chart. JavaScript libraries such as Datatables and Slick Grid are used to create interactive tabular views that allow users to rearrange and manipulate their data. By using these libraries, applying object oriented design patterns to JavaScript and adopting a data driven architecture, developers have been able to reuse components across applications, speeding up development time while reducing the reliance on testing. Every time a component is reused in an application it is another iterative test for that piece of software, making it and the applications that use it much more reliable.

RESTful Web Services

The NIF comprises of many software applications that interact with one another. In the past SOAP web services were created to facilitate this application communication. As systems grew more complex, the creation and maintenance of SOAP-based services became very time consuming. The programmer needs to create the service logic, service interface, WSDL, complex type mapping and client stub code and then make sure these pieces are in sync when modifying the service (Fig. 2).

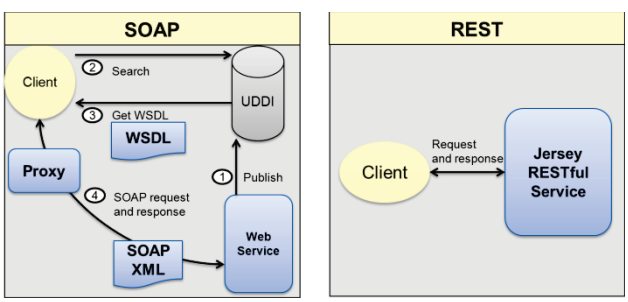

Figure 2: REST and Java Jersey allow for a simple service architecture that can return data in numerous formats for a variety of application uses.

Tools exist to help with this, but it still requires a lot of work to maintain. Because of these complexities the NIF started using RESTful services. Using Jersey, a RESTful Web Services Java framework, the NIF was able to quickly create web services that return data in numerous formats such as XML or JavaScript Object Notation (JSON). With Jersey, a single Java class can be exposed as a web service which is much easier to maintain than the multiple pieces associated with SOAP development. By switching to RESTful services, developers have been able to simplify the interfaces between applications as servers and clients can have a single service entry point.

Web GL and HTML5 Canvas

WebGL is an open API that uses HTML5 Canvas to display 3D graphics. Compatible with most modern browsers, code is written in JavaScript and works directly with HTML. As HTML5 continues to gain momentum more and more 3D JS libraries such as three.js, sceneJS, and canvas3D are being created to help abstract the details of WebGL, making it easier to add it to applications. Using WebGL the NIF created, The Target Viewer, an application that allows users to visually inspect the target in 3D. During a NIF laser shot a laser pulse is split into 192 separate beams, amplified, and directed to a BB gun pellet-sized target. Prior to an experiment it is critical that the target is in pristine condition as a small particle on the target can change the outcome for the entire experiment. The Target Viewer combines hundreds of images and data points and constructs an interactive 3D view of the target. Users can zoom, rotate, pan, and query specific sites for information and view high resolution images (Fig. 3). Using WebGL the NIF is able to confidently know the true condition of the target.

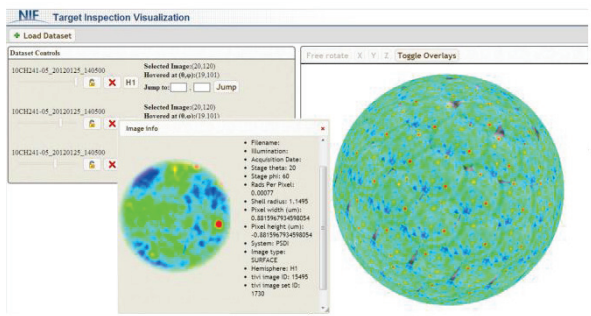

Figure 3: An interactive 3D WebGL tool that maps hundreds of images around a NIF target allowing users to zoom, rotate, pan, and query specific sites for information

Oracle Application Express (APEX)

A very common application developed for the enterprise, is the creation of a CRUD (create, read, update, delete) interface on top of a database. The primary function of these applications is the manipulation and visualization of data. Developing bespoke applications can be a significant drain on resources. NIF software engineers

ISBN 978-3-95450-139-7

selected the Oracle Application Express (APEX) framework to fulfil this need for CRUD applications. APEX is a rapid application development tool built on-top of the Oracle Database [3]. APEX provides a browser based development environment along with many different user interface tools and widgets. Application templates and themes create for a common user experience across all applications and devices, both desktop and mobile. One feature that has proved tremendously successful is the interactive reporting capabilities within APEX. The interactive reports give the end user control to customize, build, and publish their own reports [4].

Figure 4: Advanced user-driven interactive reporting with APEX along with a custom plugin augmenting the framework to achieve Excel-like fixed headers and columns.

Users can sort and filter data, select specific columns, highlight rows, create new computational columns, aggregate data, and build charts from the data (Fig. 4). This feature alone has saved countless hours, reducing development time while increasing customer satisfaction. Previously developers would spend a lot of time creating and building custom reports for users. Now users can create their own reports allowing software developers to focus on other tasks. Powered by PL/SQL and jQuery, APEX allows developers to extend upon the framework and build custom components and widgets to fit their unique business needs. This is extremely valuable to rapid development; the novice user only needs a web browser to get started and has a wide array of user interface options to choose from while the more experienced developer can extend its functionality and build custom objects to fit distinct business needs. An application at the NIF that demonstrates this is the Shot Planner, used to help schedule and plan shots. Users required two advanced features that were not directly available in APEX; Excel-like reporting with fixed headers and columns, and an advanced drag and drop calendar. To create the Excel-like fixed headers and columns, a plugin was written using jQuery that extended upon APEX's interactive reporting capabilities (Fig 4). As a plugin, this feature can now be easily reused in any interactive report. The second task was to create a complex calendar display of the NIF shot data. APEX provides a drag and drop calendar, but managing the NIF shot schedule required a much more customizable UI. Again using jQuery and APEX RESTful reports, the NIF created a custom calendar framework to fit their needs. Both the custom table and calendar components are objects that can be reused inside any other web or APEX application. Using APEX, the NIF can now deliver feature rich applications that used to take months to build, in a matter of days.

WebSockets and Node.js

The web has been built around a request/response paradigm. A client web browser requests a web page and then the server responds presenting the desired page or data. There are many times when this model falls short with the inability for the server to send data to the client in the very moment when it knows about the data. Over the years different server side "push" frameworks have emerged, but all just build on top of the request/response model using different techniques to open a connection between the server and web browser [5]. The introduction of HTML5 WebSockets changes this model as it creates a persistent connection between the client and the server allowing both parties to communicate with each other at any time. This concept of applications being able to push data directly to users and other applications the instant it becomes available is revolutionary and something the NIF wanted to use. A common problem the NIF and many other applications face is the inability to communicate directly with the currently logged in users. There are times when applications need to be taken offline for maintenance or a message needs to be sent to users of a specific application. In the past the best notification mechanism was an email sent to all known users of the given application. This approach has numerous flaws: the user might not be reading his/her email at that time or a particular user might not even be included in the email. To address this problem the NIF used WebSockets along with a Node.js server and created an application called Nodeify. Node.js is a server-side asynchronous event-driven system written in JavaScript that can run across distributed devices [6]. The ability to create scalable network applications while using the common JavaScript language made Node.js the perfect answer for real-time application communication.

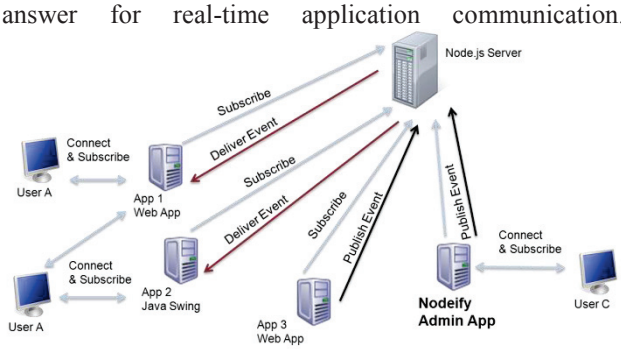

Figure 5: A Real-time event driven notification system built for scalable network applications using WebSockets and Node.js

Nodeify allows for messages to be sent to users or applications (Fig. 5). Once sent, these messages are

ISBN 978-3-95450-139-7

pushed directly to the clients without the client needing to refresh their screen. Nodeify's most notable feature is that it can easily plug into any existing web or thick client Java application. With a simple file include applications are able to push messages to other applications and clients in real-time. This universal design enabled the NIF to provide notification capabilities to all of its applications without needing to write custom code for each application, saving time and money.

SUMMARY

Web 2.0 technologies deliver many tools and frameworks that help to create dynamic and responsive applications. A strong knowledge of the latest JavaScript libraries has helped NIF developers create quick, feature-rich user interfaces. Adopting a RAD framework, like APEX, has sped up development time while enhancing user satisfaction with its custom reporting features. Applying HTML5, WebGL and WebSockets has provided a level of interaction with our users that was previously unfeasible. Identifying the proper technologies and their use in delivering application solutions has helped the NIF successfully respond to the ever-changing customer demands and business environment.

REFERENCES

[1] Heraclitus, Greek Philosopher
[2] "Market share trends for JavaScript libraries for websites"; http://w3techs.com/technologies/history_overview/javascript_library/all; http://www.jquery.com
[3] "Oracle Application Express: Overview; http://www.oracle.com/technetwork/developer-tools/apex/overview/index.html
[4] "Oracle Application Express: Interactive Reporting"; http://www.oracle.com/technetwork/developer-tools/apex/application-express/irrs-083031.html
[5] P. Lubbers and F. Greco, "HTML5 Web Sockets: A Quantum Leap in Scalability; for the Web"; http://www.websocket.org/quantum
[6] Node.js website; http://nodejs.org

ISBN 978-3-95450-139-7