Rapid Development Using Web 2.0 Technologies

Presentation to

14th International Conference on Accelerator & Large Experimental Physics Control Systems (ICALEPCS)

October 6-11, 2013

Scott Reisdorf

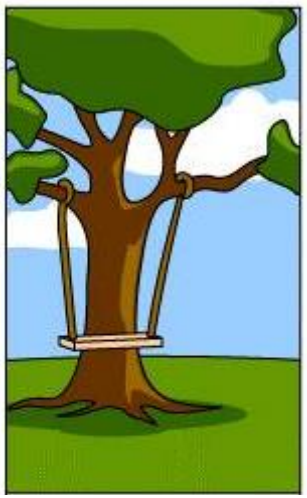
Software Engineer, LLNL

Lawrence Livermore National Laboratory • National Ignition Facility & Photon Science

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

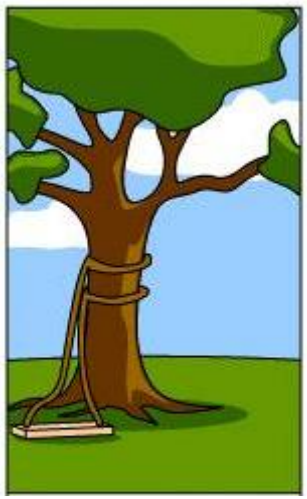
Why do we need Rapid Application Development (RAD)?

How the customer explained it



How the project manager understood it

How the analyst designed it



How the programmer wrote it

What the customer really needed

Source: http://www.paragoninnovations.com/ng4/guide.shtml

Rapid Application Development (RAD) Using Web 2.0 Technologies

- Change is Constant
  - Creating and modifying applications to business needs
  - Requirements update often with a high degree of ambiguity
  - Infrastructure changing; Java updates, the Cloud
  - Needs to integrate with operational web & Java applications

- NIF needs tools and frameworks that allow developers to quickly build and update applications and prototypes
  - jQuery, jQuery-ui
  - Oracle Application Express (APEX)
  - WebGL
  - WebSockets & Node.js

The talk will highlight some of NIF's experience with Web 2.0 technologies

jQuery & jQuery-ui
The Write Less, Do More, JavaScript Library

- jQuery is a powerful, cross-browser, feature-rich JavaScript (JS) library
- **Pros**
  - Industry standard JS library.
    - Used by 86% of top 10,000 websites[1]
  - Foundation for many web applications/frameworks
  - Versatile across most web frameworks; Code reuse
  - Open source community; many libraries, plugins, and examples
  - Easily create or extend any user interface (UI)

- **Cons**
  - Can yield to unwieldy code; MVC frameworks such as AngularJS, JavascriptMVC help
  - Tougher to debug; tools like Firebug or Eclipse VJET can help

Facilitates the creation of responsive, dynamic user interfaces
Can lead to overly complex code that is hard to maintain if you don't use modular design with an MVC framework

1: Source http://trends.builtwith.com/javascript/javascript-library

Oracle Application Express (APEX)
For Web applications based on an Oracle database

- **Create, Read, Update, Delete (CRUD) application development**
- **Pros**
  - Simple and easy to use browser based development environment
  - Many out of the box UI widgets (tables, charts, form elements)
  - Extensible framework built on top of jQuery and jQuery-ui
  - Rich reporting tools for data dissemination
  - Allows developers to focus on the data model rather than the code
- **Cons**
  - Pure PL/SQL and HTML/JavaScript development
    - No Java
  - Work within limitations of the tool/framework
  - Upgrading APEX requires all APEX applications in the database to upgrade; Oracle 12c Database addresses this issue

Can dramatically decrease application development time while delivering feature-rich UI

Shift from Java to PL/SQL can be difficult for developers

APEX Case Study: NIF Shot Planner

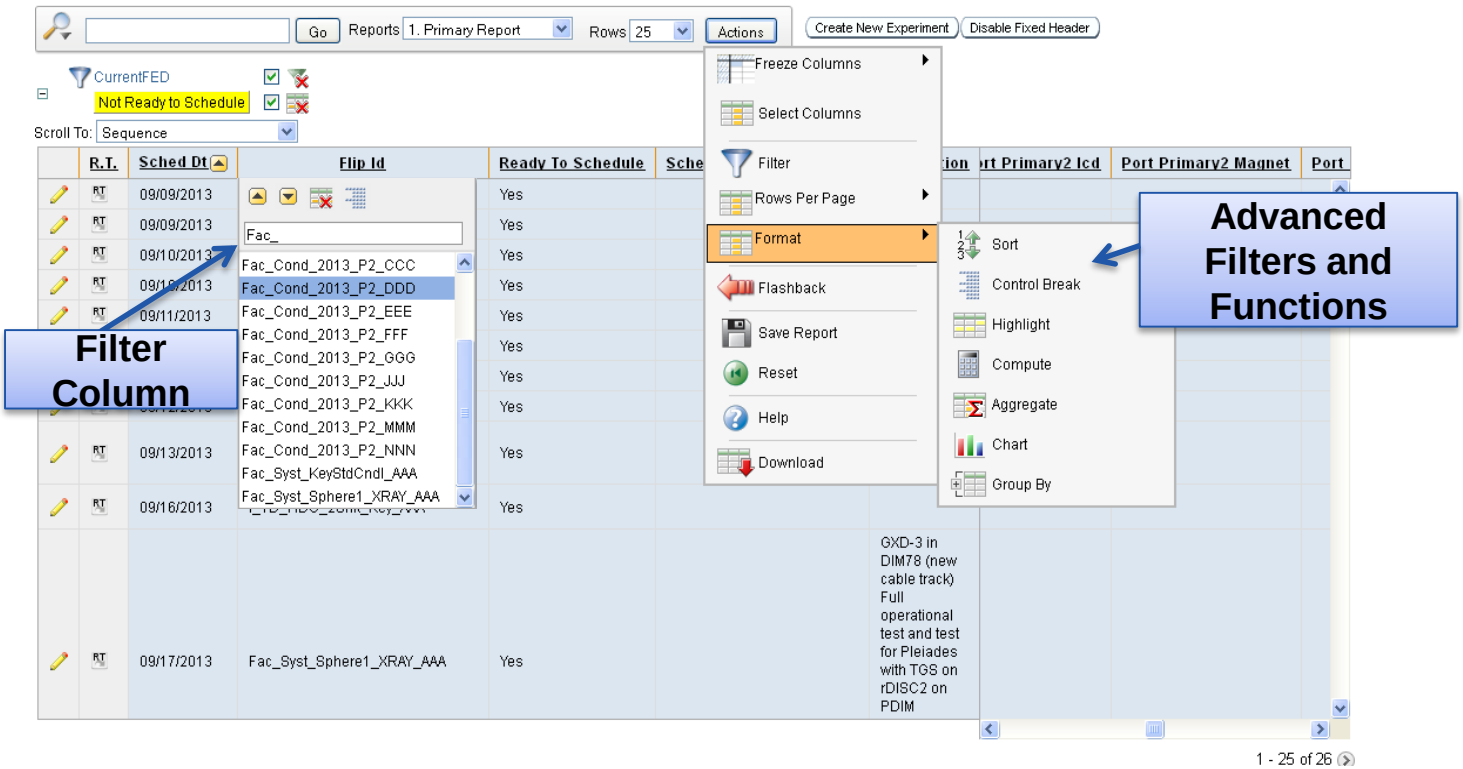
- Experiment planning application; interactive calendar and scheduling tools

JSF Application

- Development Time: 6-8 months
- Out of the box components didn't scale with large data
- Tough to extend the framework

Oracle APEX Application

- Development Time: 2 months
- Many easily extendible components
- Common look and feel across apps

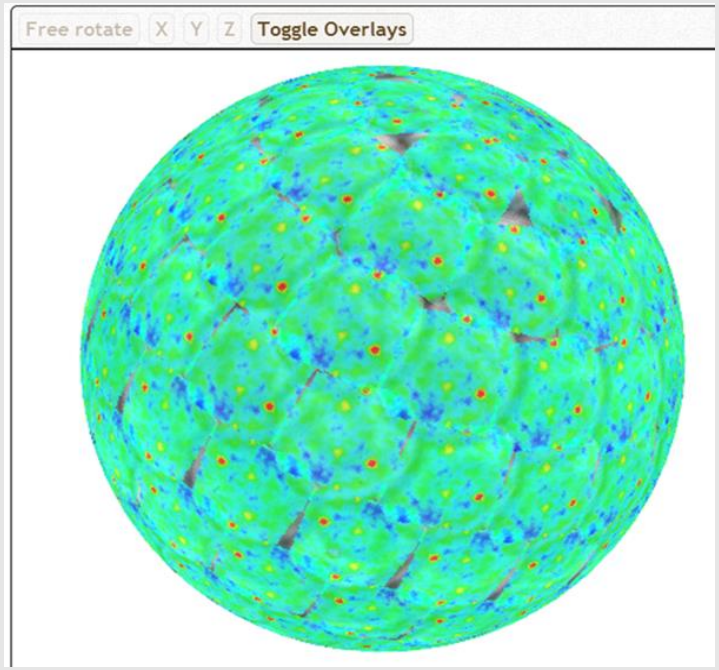


WebGL (Web Graphics Library)
JavaScript API for rendering interactive graphics

Open API based on the OpenGL ES 2.0 standard that can be used with HTML5

- **Pros**
  - — Based on industry standard
  - — Growing user base offering support and scientific solutions
  - — Impressive real time capabilities
- **Cons**
  - — WebGL only supported by Chrome, Firefox and Safari; Future support with IE11
  - — Steep learning curve with limited applicability
    - – Three.js, SceneJS can help

NIF 3D Target Viewer

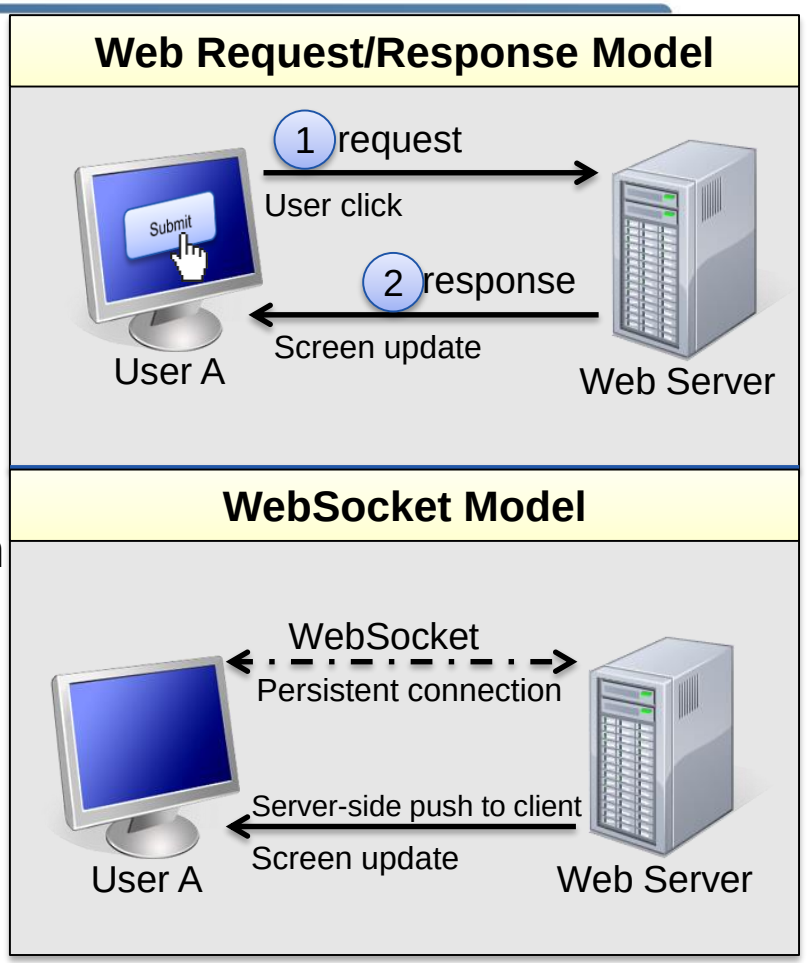


The tool maps hundreds of images around a NIF target that can be spun by the user

For specific cases WebGL can deliver impressive 3D interactive applications
Most business apps don't need this level of interactive visualization

WebSockets and Node.js
Platforms for building scalable network applications

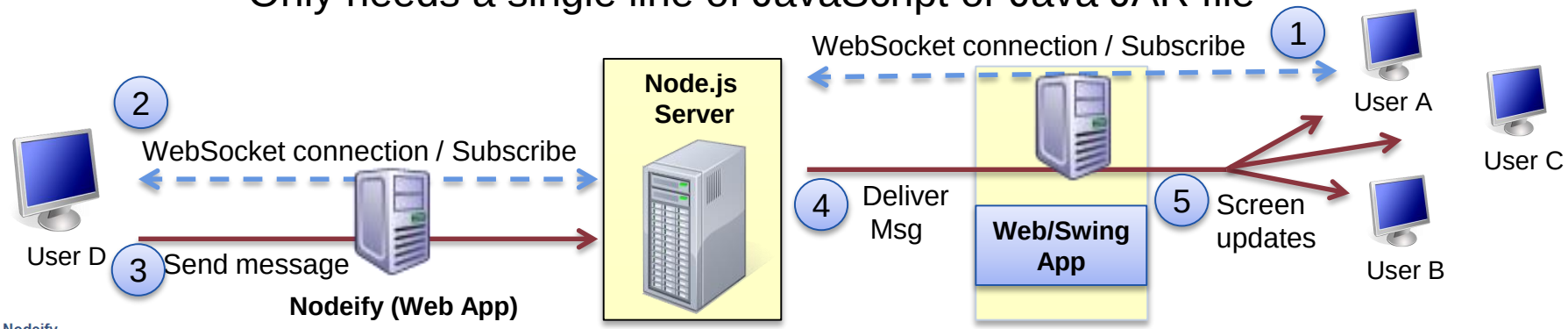
- **Traditional Web Request/Response Model**
  - — Not suitable for real-time communication
- **WebSockets**
  - — Direct communication between client & server
- **Node.js**
  - — Server-side asynchronous event-driven system using JavaScript
- **Pros**
  - — Event base I/O model; Active development
  - — JavaScript on both the front and back end
- **Cons**
  - — New technology to learn and maintain

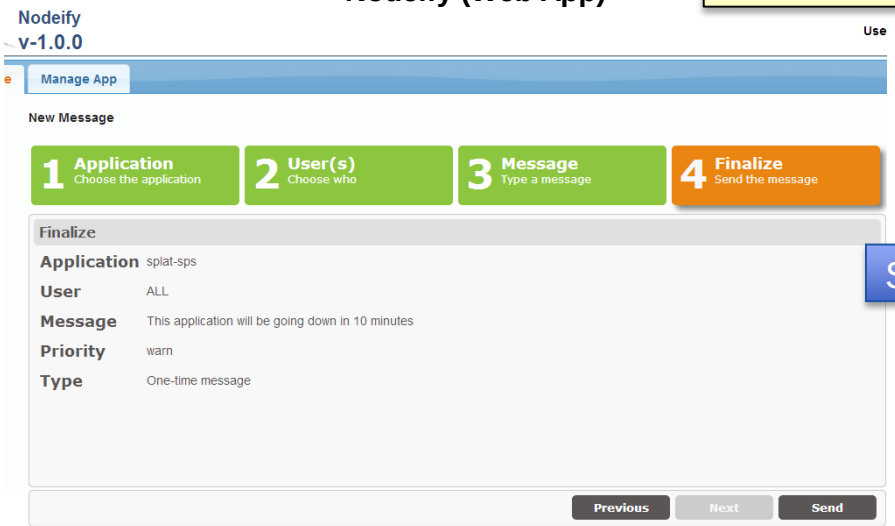


Suited for real-time apps that run on distributed devices

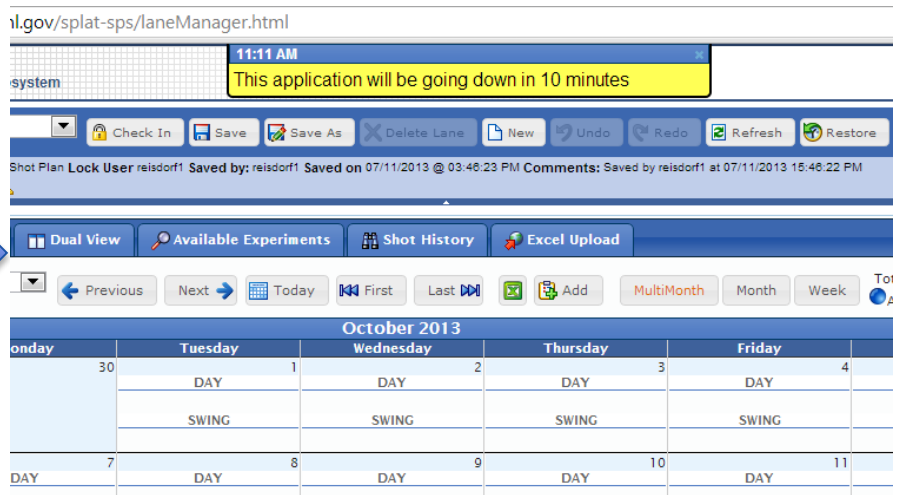
WebSocket & Node.js Use Case
Nodeify – Real-time application communication

- Nodeify
  - — Real-time notification system:
    - – Broadcast public and private messages to application users in real-time
  - — Only needs a single line of JavaScript or Java JAR file





Send Message



Nodeify Admin application; create message

User's screen auto updated with message

Summary of NIF's Web 2.0 experience

- Web 2.0 technologies can be readily adopted by experienced developers
- Features have been successfully used to speed up and simplify development
- Latest JavaScript based technologies have improved user satisfaction with rich features and quick user interfaces

Contact Information

- Scott Reisdorf
  - — reisdorf1@llnl.gov

Web 2.0 makes many promises and does deliver on them. Matching an application with the relevant technology is key


NIF