SNS Online Display Technologies for EPICS

Kay-Uwe Kasemir, kasemirk@ornl.gov
Xihui Chen
John Purcell
Katia Danilova

ORNL/SNS
ICALEPCS 2011, Grenoble, France
We Have Good Control System Displays

Control System Studio

– Portable Java Technology
– Integration of Displays, Alarm System, Archive, …

- From Anywhere
- On any device
  - and any web browser

OK, nobody asked about the toaster, yet. But it’s getting there, see www.theonion.com
Web 101: Hypertext Transfer Protocol

1. Connect to TCP port 80 on icapecs2011.esrf.eu

2. Request: “GET index.html HTTP/1.1”

3. Response: “<html>..<body>..icapecs 2011…</body><html>”

4. Disconnect
Control System 101: Events

- Subscribe to Events
- Cavity69Amp=13.2kV
- Pump47_Speed=1225 rpm
  (Pause, no changes)
- Cavity69Amp=13.3kV
- Turn pump off
  - Pump47_Speed=475 rpm
  - Pump47_Speed=123 rpm
  - Pump47_Speed=0 rpm
  (Network connection remains open)
Fundamental Mismatch

Control System

Events

– Updates can happen at any time

Web Client & Server

Request/Response

– Updates only happen after a “Reload” of the web page

Wait, that can’t be true!

There are web sites that display updates without me having to push the “Reload” button...

What about “Server Push”?
Solution 1: Web Browser Plug-Ins

• Adobe Flash Player, Apple QuickTime, …
  – Web Browser can display video

• CAML WebCA
  – Web Browser subscribes to EPICS Channel Access!

But:

1. Specific to certain web browsers. Won’t work on all your devices.

2. Firewalls likely to restrict Channel Access traffic to control system to access to plant network. Can’t get there from your phone.
Solution 2: Ajax Poll

Web Server:

JavaScript in web client performs periodic poll:

1. XMLHttpRequest
   “GET /values HTTP/1.1”

2. Response:
   “Pump47_Speed = 1234rpm, Cavity69Amp = 13.2kV”

Web Client:

3. JavaScript updates affected section of web page
   No manual reload!
SNS Status Web Pages

As of Thursday, September 01, 2011 14:09:11
Manual Beam Switch Turned On
MPS Allows Beam
Continuous Shot Mode
To: Target, MPS: 1 mSec
Power at: 680.91 kW

As of 15:28:00 on 09/01/2011, Reactor Power is at 0 MW
The reactor is currently shutdown for the end-of-cycle 437 refueling outage.
Startup for cycle 438 is currently planned for Monday, October 10.

SNS Energy Plot

Accelerator Messages
Accelerator Beam is on target at 700 kW.

CP 1B repair
2011-08-22
14:16
Upon completion of the RFQ Cryo Pump 1B repair/replacement, please type the following in an open terminal window (accl-oper privileges are all that's necessary): [accl-oper@ics-ops-ccr11 ~]$ caput /dev/agi1 rsn_summary CP INPB
"RFQ_Vacuum__MS" This restores the status indicator for this device to the start map geographical summation indications.
Status Web Technology

- JSP (Apache Tomcat)
- Web Server subscribes to predefined list of EPICS Channels, keeping track of ‘current’ values
- Ajax periodically updates sections of web page

EPICS

Subscribe  ... Events

Tomcat
BeamPower=680.91 kW
Mode=Target
Rate=1 mSec

Request  Response

Web Browser

As of Thursday, September 01, 2011
14:09:11
Manual Beam Switch Turned On
MPS Allows Beam
Continuous Shot Mode
To: Target, MPS: 1 mSec
Power at: 680.91 kW
Status Web Page Summary

✓ Very robust
✓ Works on every web client
✓ Minimum CPU and network requirements
  ○ Content is pre-defined (Overview, Beam Info, …)
    – Cannot add specific section for each user
  ○ Only slow periodic updates
    – some 3 sec, others ~30 sec
Solution 3: Long Poll

Ajax Poll with special timing

- Client sends request
- Server delays response until **new data** becomes available
- Server returns data
- Client handles data and **immediately starts new request**

### Plain Ajax Poll

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request</td>
<td>Response</td>
</tr>
<tr>
<td>Response</td>
<td>Request</td>
</tr>
</tbody>
</table>

### Long Poll

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request</td>
<td>Response</td>
</tr>
<tr>
<td>Response</td>
<td>Request</td>
</tr>
</tbody>
</table>
Using Long Poll...

- **Looks like event-driven display to end user**

- **But:**
  - Slower than pure event mechanism that would not need requests
  - Java Script code needs to handle differences between web browsers
  - Number of concurrent Ajax requests is limited (2, 6, .. depending on web browser)
    - Cannot have one Long Poll per data point
    - Better have only one Long Poll for the whole web page

Need to be familiar with HTML, DOM, JavaScript, Style Sheets as well as server-side Java
SNS Dashboard

Users log in/out

Can display any process variable, updating as the data changes

Users can add/move/configure widgets as desired.
Dashboard Technology

- **JSP (Tomcat)**
  - Loads web page configuration for user from RDB (MySQL)

- **Java Script in Web Browser**
  - Sends “Subscribe” request for each widget to Web Server

- **Java Script for web page**
  - Performs Long Poll for updates

- **Web Server returns data as it arrives from Control System**

---

**EPICS**

Subscribe up

... Events

**Tomcat**

PVs for current user:

```
BeamPower=680.91 kW
Mode=Target
Rate=1 mSec
```

Subscribe up
dashed
don't care

Long Poll for updates

---

<table>
<thead>
<tr>
<th>Process Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCL Amp</td>
</tr>
<tr>
<td>Energy</td>
</tr>
<tr>
<td>Mode</td>
</tr>
<tr>
<td>Again</td>
</tr>
</tbody>
</table>

---

15 Managed by UT-Battelle for the Department of Energy
JQuery

• Java Script Library

• Simplified the code
  – Common API across different web browsers
  – Web Client DOM manipulation
  – Ajax calls
  – “Drag” widgets inside web browser to arrange them
  – Graphical gimmicks
Dashboard Details

- Web Clients will subscribe
  - Web Server subscribes to Control System PVs…

- but they don’t cleanly unsubscribe
  - Web Server’s PV pool would grow…
  - Use time outs to delete client resources

- Servlet “GetUpdates” called in Long Poll
  - Returns right away with accumulated events
  - Otherwise waits a second, checks again, …
  - .. for up to 10 seconds, then returning empty response
    - If waiting much longer, the server/client web session times out
Dashboard Summary

✓ Works on every web browser

✓ Faster updates than Status Web
  – Updates to cell phones can be intermittent

✓ Users can create their own page
  – Picking from the list of available widgets

○ Higher CPU and Network load
  – A lot more Java Script, always in Long Poll

○ Distributed Java on Server, JavaScript/DOM in web browser hard to debug
Solution 4: Eclipse RAP, the Theory

Eclipse RCP  
(Rich Client Platform)

- Java Application code for Ctrl. System Display
- SWT

Standalone Application for Windows, Linux, OS X

Eclipse RAP  
(Rich Ajax Platform)

- Same(!) Java Code
- RWT

Web Page Java Script RWT Client

Web App, installed in e.g. Tomcat

Web Browser (via HTTP)

- RWT library has (almost) same API as SWT
- Instead of creating widgets on local display, it sends information to web browser
  - Updates via Long Poll, but handled by RAP/RWT
Eclipse RAP in Practice

CSS ‘BOY’

WebOPI
WebOPI Details

Started with CSS ‘BOY’ code.

Differences:

- SWT directly updates local display
- ‘Advances’ graphics (Gradients, …)
- One (local) display
- RWT queues JavaScript for Long Poll
- ‘Normal’ graphics
- Many (virtual) displays (one per web client)

Main impact on code, otherwise “single source”
WebOPI Summary

✓ Allows us to leverage both existing CSS BOY code and display files

✓ Users can create any type of display they want and see it on the web

✓ Works with many web browsers

✓ Don’t have to deal with HTML, JavaScript code
  ○ Relies on pre-release versions of RAP code
  ○ CPU and network use naturally higher, because it has to use Long Poll and JavaScript instead of direct display updates
Summary

- Users want web access to the control system
  - But HTTP is meant for request/response, not subscription to events

- Web pages with Ajax
  - Easy on resources, very portable, but not very dynamic

- Long Poll, a lot more JavaScript
  - Needs more CPU, harder to implement, but more dynamic

- Eclipse RAP
  - Needs even more resources, but fully dynamic
  - Re-use existing Java code, no worries about details of JavaScript