AJ AX Performance and Monitoring

Ron Bodkin, Glassbox Leader
ron.bodkin@glassbox.com
Agenda

- AJ AX Overview
- AJ AX Technology
- Challenges
- Solutions
AJ AX...

Google Gmail Calendar more

Compose Mail

Inbox (1757)
Starred
Chats
Sent Mail
Drafts
All Mail
Spam (549)
Trash

Contacts

Welcome Sign Out, Welcome

Gmail

Google Alerts

Yahoo! Mail

$699 Notebook
From Dell™ Home

Email

Dell

Inbox (180)
Drafts (1)
Sent
Spam (1449)
Trash

Close

Compose

Find Messages...

Reply

Forward

Move

Print

Spam

Delete

More Actions

View

Check Mail

Compose

Reply

Forward

Move

Print

Spam

Delete

More Actions

View

Dell's 22nd Anniversary - Special thanks withThu, 5/18/06 10:45 AM

Robert David
TRUST AND UNDERSTANDING
Fri, 5/19/06 7:09 PM

Churchill Club
The Churchill Club: VC-What's Hot/Wh Fri, 5/19/06 1:03 PM

Churchill Club
The Churchill Club: VC-What's Hot/Thu Tu, 5/18/06 11:45 AM

Dell Direct Deals
Dell's 22nd Anniversary - Special thanks witt Thu, 5/18/06 10:57 AM

AMERITRADE CLEARING
INX INC. Annual Meeting Tue, 5/16/06 11:03 PM

eNews@baybreakerx.cor
ING Bay to Breakers - Last call to get Tue, 5/16/06 12:25 PM

support@oneworldhosting.com
News from One World Hosting: Reselll Mon, 5/15/06 12:06 PM

Dell's 22nd Anniversary - Special thanks with 22% savings
Dell Direct Deals <dell@homeandhomeoffice.dell.com>

To: rjboedkin@yahoo.com
Asynchronous JavaScript And XML

A group of technologies: HTML, JavaScript and XML

Still HTML-based, but with JavaScript packages that make web pages feel more responsive

Parts of a page fetch data and update without the major re-load everyone is used to

Used correctly, Ajax brings the interactivity of good desktop apps to the Web, such as a spell checker that checks on the fly like Outlook.

Ajax can be chatty, so it needs a good network and thoughtful design
Other Options

- Fat Client
  - Most interactive
  - Hard to distribute, maintain

- HTML
  - Easy to write, most portable
  - Less interactive especially on WAN

- Flash
  - Wide deployment (>95%)… still mostly used for animation
  - Harder to develop, integrate

- Java
  - Applets had early promise, limited to a niche

- Microsoft XAML/Avalon
  - Windows Vista only… ask me in 2009
Why AJAX?

- Rich interactivity
  - Responsiveness like a desktop app
  - Lively site
  - Ease of use

- Browser-based
  - Standards
  - Ubiquitous
  - Easy update

- Productive, usable, networked applications
Why Now?

- Technology matured
- Successful large-scale applications
- Emergence of tools and frameworks
- Even as IE went from 95% to 85% share
New Interaction Patterns

classic web application model (synchronous)

Ajax web application model (asynchronous)

Source: Adaptive Path
Agenda

- AJ AX Overview
- AJ AX Technology
- Challenges
- Solutions
Enabling Technologies

- XMLHttpRequest
  - Allows *asynchronous* communication with server
  - Generates events as data is received

- JavaScript
  - Better portability across major browsers
  - Still a lot of conditional logic required

- Dynamic updates to page
  - DOM gives full object access
  - innerHTML attribute for modifying HTML inside a tag
Javascript (Browser) Libraries

- dojo (widgets, packaging, utilities, persistence)
- Prototype (framework)
- script.aculo.us (effects, widgets)
- AjaxTK/Apache Kabuki - Zimbra (widgets)
- Yahoo! User Interface Library (widgets)
- TIBCO General Interface (framework & tools)
Browser + Server Frameworks

- **Java**
  - DWR (Direct Web Remoting)
  - Google Web Toolkit
  - ZK
  - AjaxTags
  - Ajax4J SF

- **.NET**
  - Atlas
  - Ajax.NET
  - MagicAjax.NET

- **PHP**
  - Symfony
  - AjaxAC

- **Ruby on Rails**

- **Python: Django**

- **Perl**
  - Catalyst
  - CGI::Ajax
Example: DWR 1.0 AJAX

```
function eventHandler()
{
    AjaxService.getOptions(populateList);
}

function populateList(data)
{
    DWRUtil.addOptions("listid", data);
}
```

```
public class AjaxService
{
    public String[] getOptions()
    {
        return new String[] { "1", "2", "3" };
    }
}
```

Source: Getahead
Reverse AJAX: Automatic Updates

Server

Event

Update

Browser

Script

?
Reverse AJAX: Polling

Server

Event

Update

Browser

Script
Reverse AJAX: Comet

- Server
  - Event
  - Update

- Browser
  - Start
  - Script
Reverse AJAX: Piggy Back

Often discussed, but who is implementing it?
Widgets, *e.g.*, Google Web Toolkit

![Widgets Examples]

- **Table**
  - Sender: markboland05
  - Email: mark@example.com
  - Hollie Voss: hollie@example.com
  - boticario: boticario@example.com
  - Emerson Milton: emerson@example.com
  - Healy Colette: healy@example.com
  - Brigitte Cobb: brigitte@example.com
  - Elba Lockhart: elba@example.com

- **TabBar**
  - Tab: 1634 1640 1642 1662

- **DialogBox**
  - About the Mail Sample

- **PopupPanel**
  - Richard Feynman: richard@example.com
Direct Service Integration

Mash-ups...

- Web page
- AJAX engine
- Order Service
- Google Maps
- Credit Agency

AJAX engine
Indirect Service Integration

Web page → AJAX engine

AJAX engine → Aggregation server

Aggregation server → Order Service

Order Service → Google Maps

Order Service → Credit Agency
Proxy Web Service Integration

- Web page
- AJAX engine
- Proxy server
- Order Service
- Google Maps
- Credit Agency
Agenda

- AJ AX Overview
- AJ AX Technology
- Challenges
- Solutions
AJ AX Problem Areas

- Excess server requests
- JavaScript problems
- Server load
- Service fragility
- Integration server problems
Excess Server Requests

- Adds complexity
  - One single action can generate a flood of requests
  - Or a handful of slower ones
  - What happens if some are unreliable?
  - One missing timeout can produce a huge backlog

- Hard to Monitor End User Performance
  - Initial page load time isn’t the issue…
  - Can’t determine cause of requests without page analysis

- Hard to correlate Requests with Pages let alone User Actions
JavaScript Problems

- Not as fast as compiled languages!
- Processing large data sets is hard
- Engines can do a lot of work
- Browser portability is still an issue
  - May require digging into framework code
- Stateful UI’s: what interactions led to errors?
  - Debugging nested DOM/CSS/JS…
- Accessibility
- HTTP limit of 2 simultaneous connections per server by default (configurable)
Server Load: Pre-AJAX

- Short spikes of activity
- Long pauses of inactivity “think time”
- Efficient for use of server sockets and threads
- Can tax CPU
- Dynamic pages integrate big static chunks
Server Load: AJAX

- Initial download of engine (static content)
- Bursts of activity (change of context)
- With more frequent updates
- Typically shorter less intensive interactions
- Frequent requests and polling use network, CPU, sockets, and threads
- Comet instead uses much more **sockets** and typically **threads** and **buffers**
- More demanding of client too
- More *static* content
Service Fragility

- Each service is a possible point of failure
  - Outages
  - Configuration problems
  - Changes to service

- Cascading slowness possible
  - Not meeting SLA
  - Tests robustness/race conditions in browser
Integration Server Problems

- Slow database query
- Too many queries
- Bottlenecks from aggregating many services
  - memory, sockets, …
- Contention for locks in server…
  - lots of threads for one session
- Outages/misconfiguration
- …Traditional problems in a new context
Typical Application Problems

Hardware Problem
Agenda

- AJ AX Overview
- AJ AX Technology
- Challenges
- Solutions
The Performance Lifecycle

- **Define**
  - Service level

- **Design**
  - Load test architecture
  - Create budget

- **Develop**
  - Code & debug
  - Unit tests

- **Test**
  - Load test often
  - Multi-platform

- **Operate**
  - Monitor
  - Update

- **Troubleshoot**
Architecture Is Key

- **AJ AX shuffles the deck**
  - Changed workloads
  - Emerging, immature technologies

- **Allocate work sensibly**
  - Informs basic goals and SLA’s

- **Budgets for both latency and scalability**
  - Interactive events (heavy & light)
  - Update callbacks

- **Prototype basic elements**
  - Many new technologies at play

- **Extrapolate to model performance**
Server Options

- The usual suspects...
  - Java EE (Tomcat, JBoss, Jetty, GlassFish, WebSphere*, WebLogic*)
  - .NET for Windows*
  - LAMP

- Emerging networking options
  - Grizzly event-based HTTP Connector (in GlassFish)
  - COMETd (Perl)
  - Twisted (Python)
  - Scala, Erlang/OTP (Concurrency Oriented Languages)
  - Apache 2 Event MPM
  - Continuation support…

* Indicates cost to use.
Emerging Standards?

The Jetty open source Java Web container:

- Uses NIO to only use threads when processing I/O
- Server requests set up a *continuation*
  - If waiting for data, they suspend by throwing
  - When data is available, they are resumed
  - The server replays the processing, but the next time they will continue
- Continuations allow using one thread per active request
- Also uses NIO split buffers to limit buffers for many requests
- DWR 2’s Reverse AJAX
  - works with Jetty continuations
  - Supports polling & COMET
Continuations

- Object to represent the state of program execution
  - Native support in dynamic languages like Ruby, Smalltalk
  - Java options rely on specialized state

- Jetty implementation supports AJAX scalability
  - Requires no side effects in code
  - REPLays state based on request state & session state
  - Doesn’t checkpoint session state…

- Other frameworks use to simplify conversations
  - RI FE framework for Java
  - Seaside framework for Smalltalk
IDE Support

- Eclipse
  - ATF (editor, debugger)
  - MyEclipse * (editor, code assist, debugger)
  - Aptana * (code completion)
- IntelliJ IDEA* (refactoring editor)

* Indicates cost to use.
Browser Development Tools

- **Debugging**
  - Mozilla Venkman debugger & profiler
  - Microsoft Script Debugger for IE

- **Inspectors**
  - WebDeveloper toolbar for Firefox: CSS, inspector
  - FireBug console viewer, inspector, debugger
  - Mouseover DOM Inspector

- **Request monitoring**
  - FireBug
  - GreaseMonkey: XMLHttpRequest Tracing/Debugging
  - Eclipse ATF

- **Logging**: MochiKit, dojo… *server-side?*

* Indicates cost to use.
Venkman Profiling Output

33 <http://localhost:8080/glassbox/dwr/util.js>
util.js: 750 - 2500 milliseconds
  Function Name: anonymous (Lines 294 - 375)
  Total Calls: 6930 (max recurse 0)
  Total Time: 2093.01 (min/max/avg 0/10.02/0.3)
  Time (ex. calls): 1011.45 (min/max/avg 0/10.02/0.15)

35 <http://localhost:8080/glassbox/js/troubleshooter.js>
troubleshooter.js: 750 - 2500 milliseconds
  Function Name: loadRowInfo (Lines 78 - 83)
  Total Calls: 99 (max recurse 0)
  Total Time: 2293.3 (min/max/avg 20.03/40.06/23.16)
  Time (ex. calls): 0 (min/max/avg 0/0/0)
Unit Test

- Baseline test for functionality AND performance

- In browser
  - Script.aculo.us, JsUnit, ASTUce

- JavaScript unit tests
  - Rhino, MS Windows Script Host
  - JavaScript Coverage Validator (beta)*

- On server
  - JUnit, NUnit
  - Java Coverage: Emma, Cobertura, Clover*

* Indicates cost to try or use.
System Test

- Functional system tests (in browser)
  - Script recording and playback validation
    - Selenium, Squish/Web*
  - Script code objects (e.g., for JUnit)
    - Waitij/Waitir (Win/IE scripting for Java/Ruby)

- Load and Stress tests: simulate traffic
  - OpenSTA (http/s only... confused by gmail)
  - JMeter (http/s only)
  - LoadRunner (http/s, script objects too)*

* Indicates cost to try or use.
Configuration

- Tune based on Load Test
- Network
  - Load balancers, firewalls, routers
- OS level
  - Maximum sockets
  - Maximum threads
- Web server
  - Timeouts
- Application server
  - Threads
  - Timeouts
- VM
  - Memory settings (e.g., -Xss thread stack size)
Systems Monitoring

- Network-level
  - Router, sniffer metrics: throughput, latency
  - End-user response time (not just http pings!)
  - Customer Experience Management complicated: Keynote*, TeaLeaf*

- OS-level: sockets, IO, processes, CPU…
  - netstat, top, Nagios, Hyperic, HP Openview*, Tivoli*

* Indicates cost to try or use.
Using Fiddler: Mail Login & View

Yahoo Mail Beta

- Request Count: 52
- Bytes Sent: 61,069
- Bytes Received: 165,677

- US West Coast (DSL - 30KB/sec)
- Round trip cost: 5.20s
- Elapsed Time: 12.20s

Gmail

- Request Count: 48
- Bytes Sent: 55,443
- Bytes Received: 51,044

- US West Coast (DSL - 30KB/sec)
- Round trip cost: 4.80s
- Elapsed Time: 7.80s
App Server Monitoring

- Container
  - Server JMX: request queue, pools, throughput
- Java VM with Java 5 JMX Data
  - All threads, memory
- Key elements
  - threads, memory, request queues, throughput, etc.

* Indicates cost to try or use.
App Monitoring with Aspects

- Aspects run automatically at well-defined points at runtime
- **No need to instrument code**
- Allows low overhead tracking
- Easy to update monitoring policies
  - Enable and disable, even sampling

Standardized support
- AspectJ load-time weaving avoids changes to build process
- Popular extensible language for Java 5 javaagents
- Spring AOP allows proxy-based option for coarse-grained components

Flexibility
- Reuse open source monitors for common APIs
- Easy to extend for custom monitoring
Glassbox Open Source

- Java 1.4+
- Discovers and tracks operations as they execute
- Load-time weaving
- Low overhead
- Detects common problems, e.g.
  - AJAX latency, load
  - Excess queries
  - Slow/broken Web services

Diagram:
- Browser
- Client Machine
- Web Server
- HTTP
- Glassbox Application
- Java Application
- Glassbox Agent
- Java Virtual Machine
- AOP
- Application Server Machine
- Web Services
- Any Database
Troubleshooting Tools Provide

1. Aggressive data filtering can reduce the data haystack early

2. Troubleshooting intelligence means users need not be experts in the tool or have written the system

3. Cleaner data and more accurate diagnoses streamline communication

Benefits

- Fix problems faster and cheaper by streamlining communication.

- Fix more problems by enabling non-experts to track them down.
Bringing It Together

DEMO
Conclusions

- AJAX is now ready for prime time
  - Emphasize high-value interactivity
- Supporting technologies becoming mainstream
  - But test carefully and watch for shifts
- Focus on architecture up front
  - Benchmark latency and throughput
- Integrate monitoring and troubleshooting up front
- We’re looking for collaborators to build better AJAX monitoring & troubleshooting
- Please leave your business card to get updates
- These slides will be updated on the post-conference CD
Resources

- **Ajaxian** for News:  [www.ajaxian.com](http://www.ajaxian.com)
- **Ajax Patterns** for Technologies:  [ajaxpatterns.org](http://ajaxpatterns.org)
- **Scaling Connections for AJAX with Jetty 6:**  [www.mortbay.com/MB/log/gregw/?permalink=ScalingConnections.html](http://www.mortbay.com/MB/log/gregw/?permalink=ScalingConnections.html)
- **Not There Yet: COMET with Apache and Jetty.**  [http://blogs.pathf.com/agileajax/2006/05/not_there_yet_c.html](http://blogs.pathf.com/agileajax/2006/05/not_there_yet_c.html)
- **What I Didn’t Know About XHR:**  [http://www.oreillynet.com/xml/blog/2006/10/what_i_didnt_know_about_xhr.html](http://www.oreillynet.com/xml/blog/2006/10/what_i_didnt_know_about_xhr.html)
- **Glassbox:**  [www.glassbox.com](http://www.glassbox.com)