Ajax Best Practices

south florida
COLDFUSION
USER GROUP

Who's The Rob?

- Name: Rob Gonda
- Codename: The Rob
- Location: Atlantis - Aqua-base One
- Employer: iChameleon Group (CTO)
- Journal: Ajax Developers Journal (E-in-C)
- Access: Group Level 7

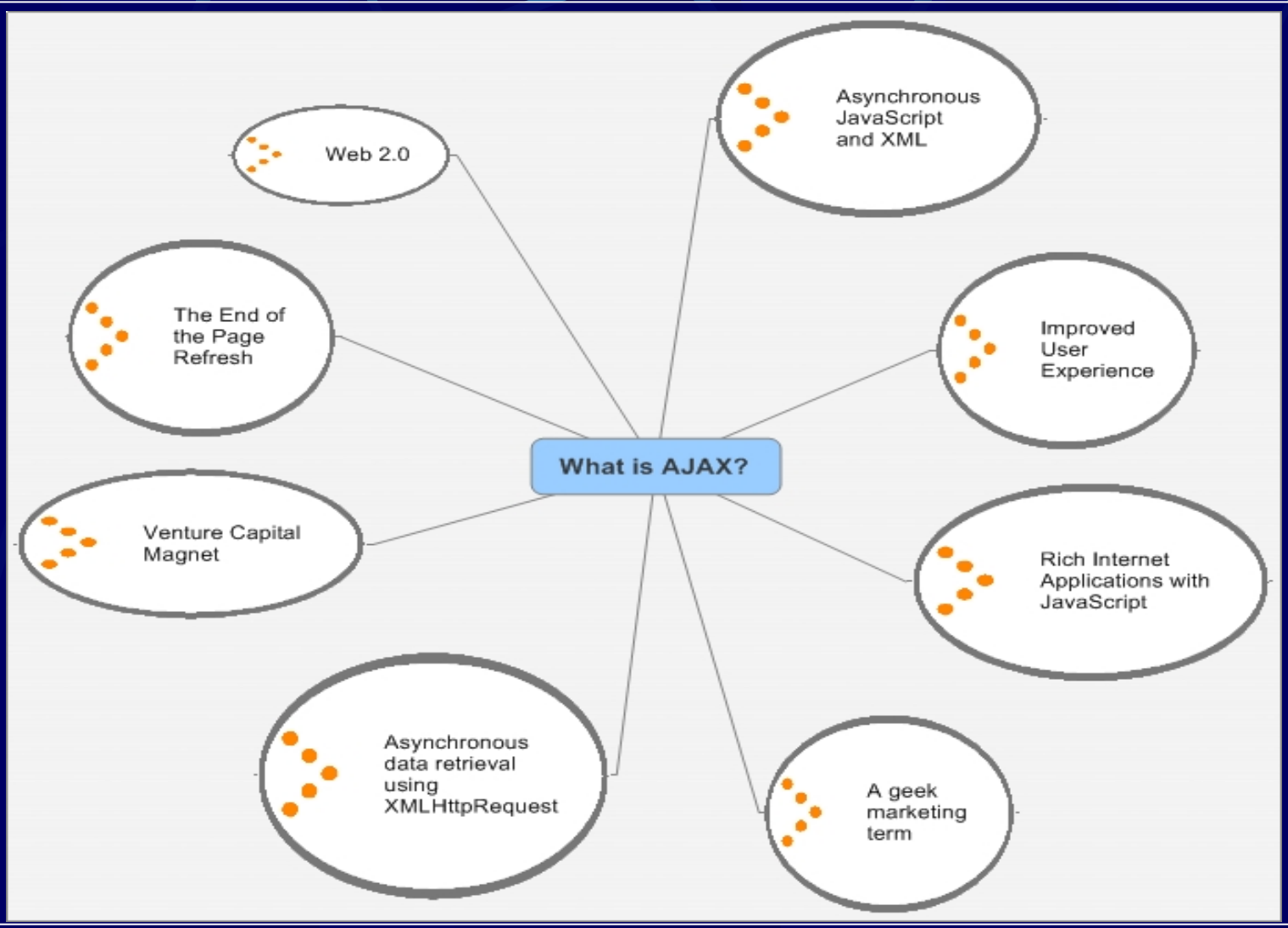
TOC

- AJAX Definition
- Ajax Vs. Classic Web
- Potential Problems and Security Rules
- Business Logic Importance
- Data Transport Vs. Instructions Transport
- Tips and tricks
- Debugging
- Design Patterns :: MVC
- Libraries, frameworks :: Model Glue / ajaxCFC / factory methods
- Live Examples
  - Simple Ajax Echo
  - Model Glue Ajax Echo
  - Model Glue Contact Manager
  - Model Glue Ajax / XMLSockets Contact Manager

!TOC

- Accessibility
- Commercial Frameworks
- ….Accessibility

What is Ajax?


What is AJAX?
Web 2.0
Asynchronous JavaScript and XML
The End of the Page Refresh
Improved User Experience
Venture Capital Magnet
Rich Internet Applications with JavaScript
Asynchronous data retrieval using XMLHttpRequest
A geek marketing term

What is Ajax?

AJAX != DHTML

What is AJAX?

- Asynchronous JavaScript + XML
- Interacting with the server without refreshing the page
- Rich Internet Applications with JavaScript
- A geek marketing term
- Venture Capital Magnet

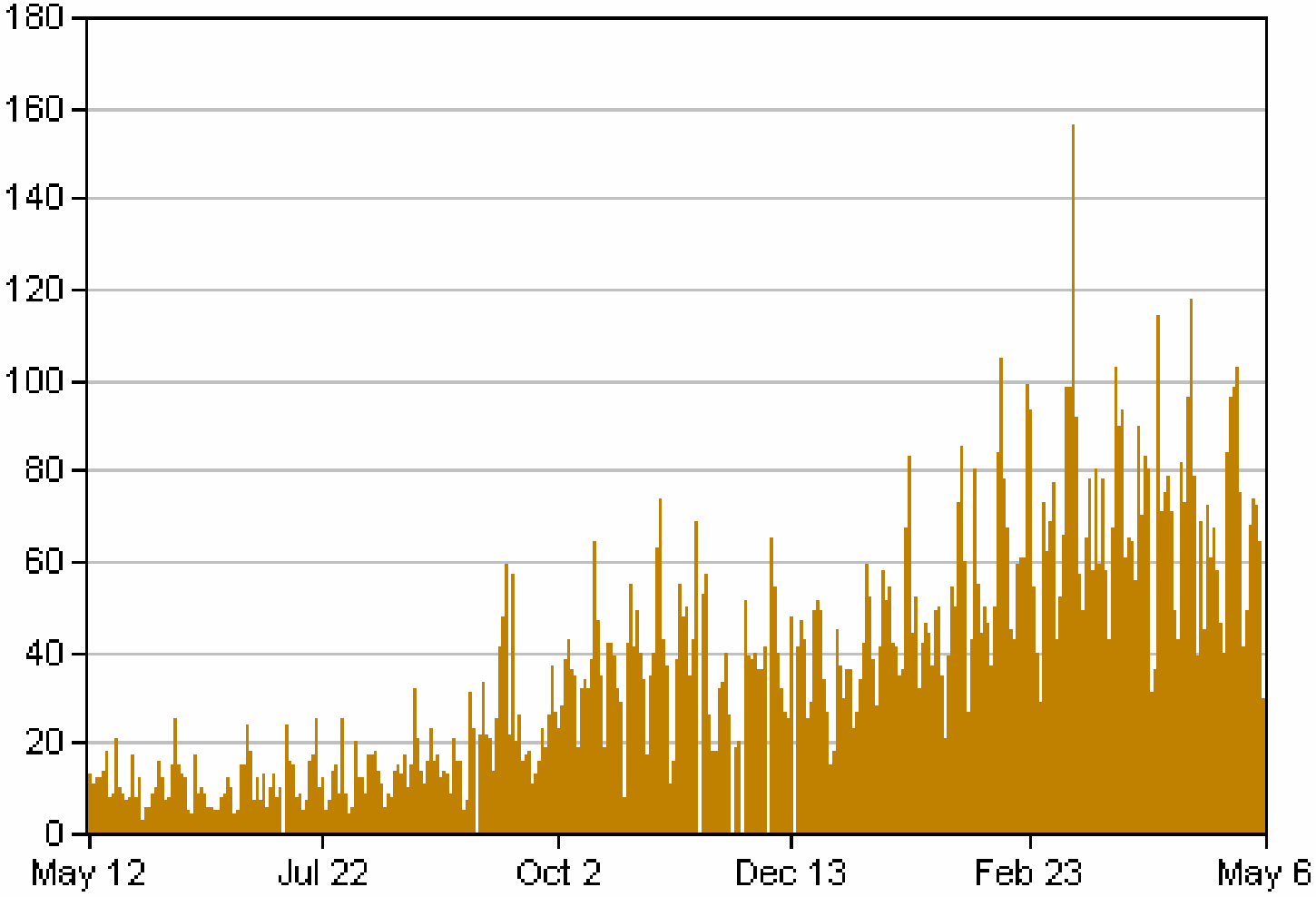
What is AJAX? (tech view)

- Asynchronous data retrieval using **XMLHttpRequest** (or not)
- Data interchange and manipulation using *XML* (or not)
- Dynamic display and interaction using Document Object Model (DOM) (or not)
- JavaScript binding everything together
- Flash Remoting for JavaScript? Not really, but close

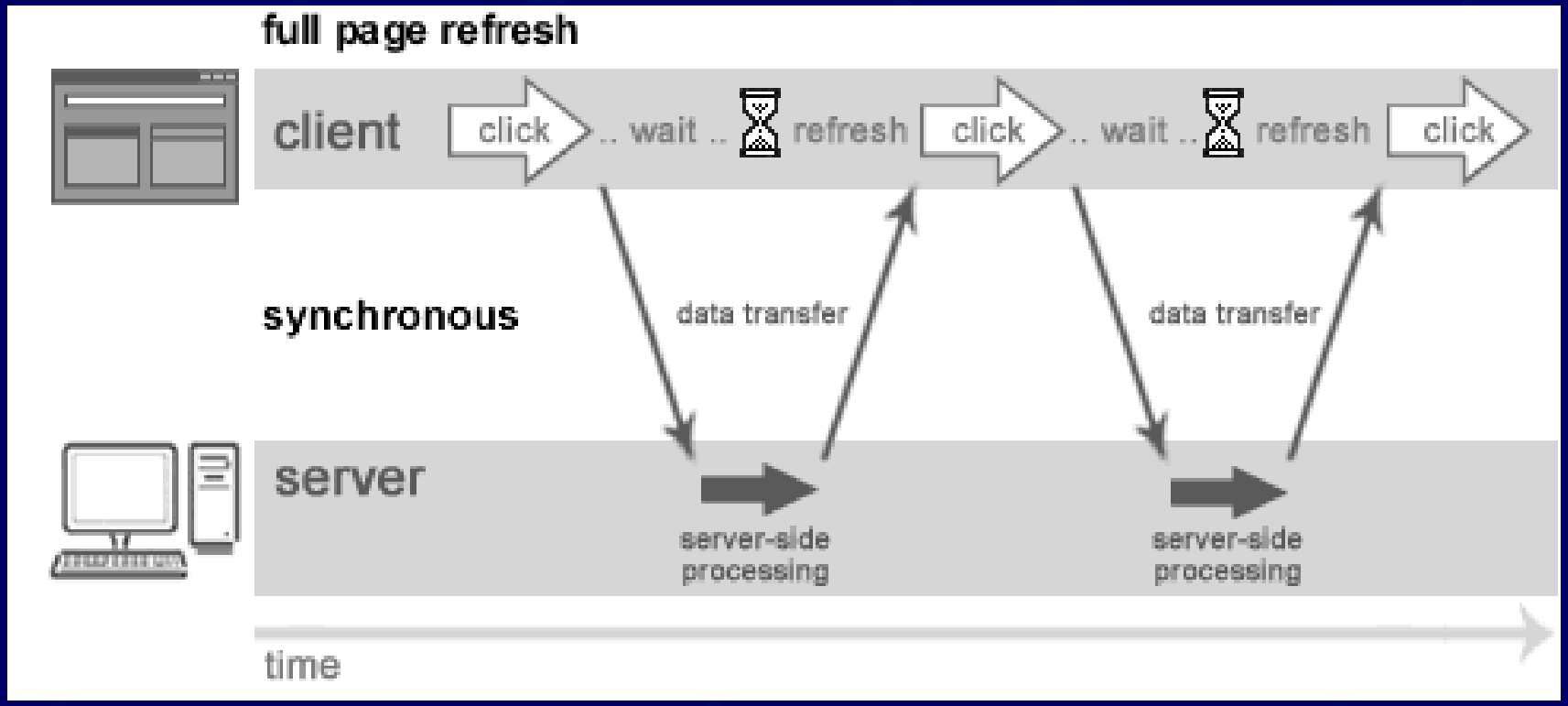
Two Types

- Ajax On The Edges
  - Ajax Widgets
  - Enhance your current site
  - Add usability
  - Degradable
- RIA
  - Single-page applications
  - Replacement for Desktop Applications

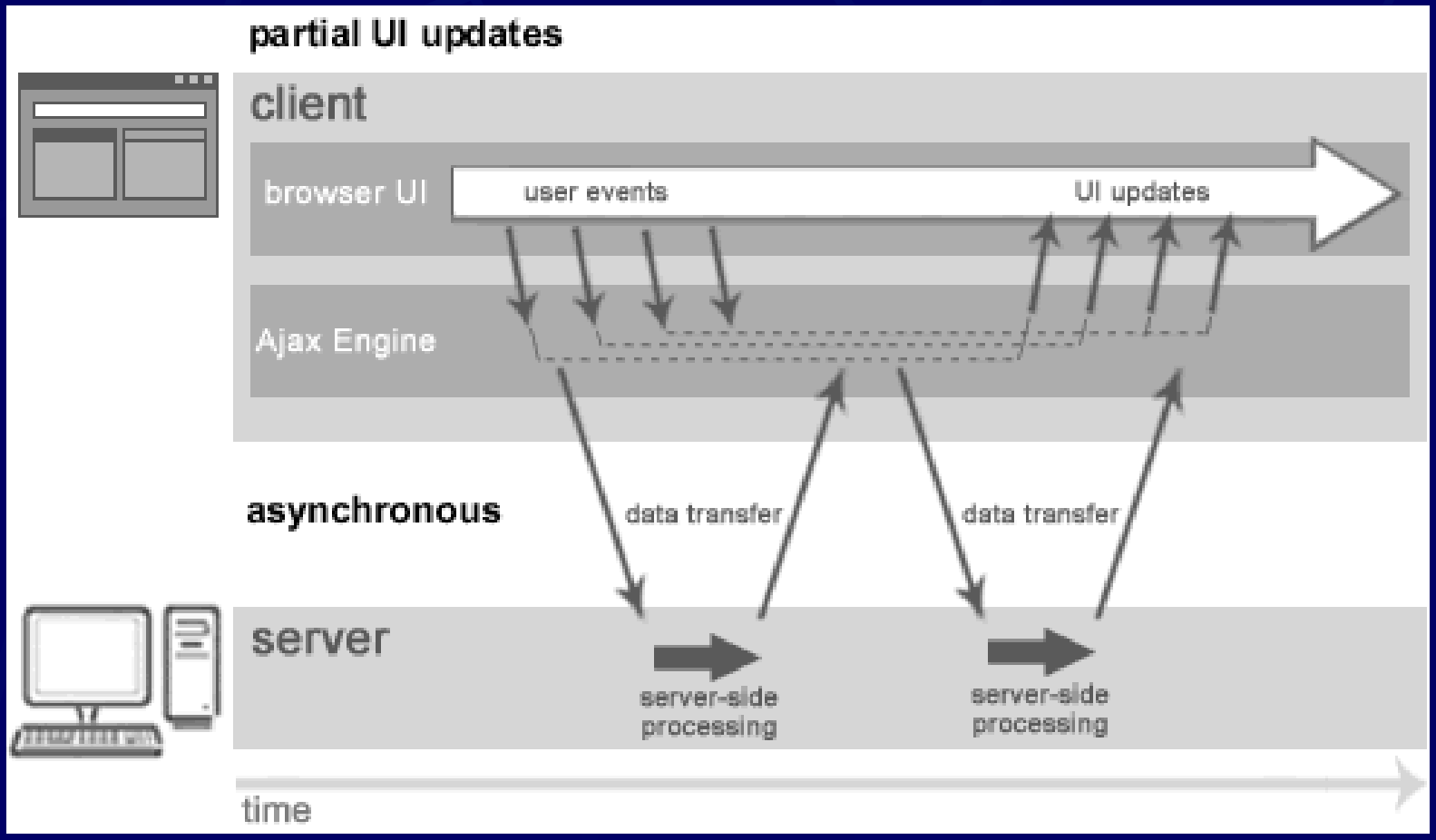
The Ajax Buzz



Classic request flow

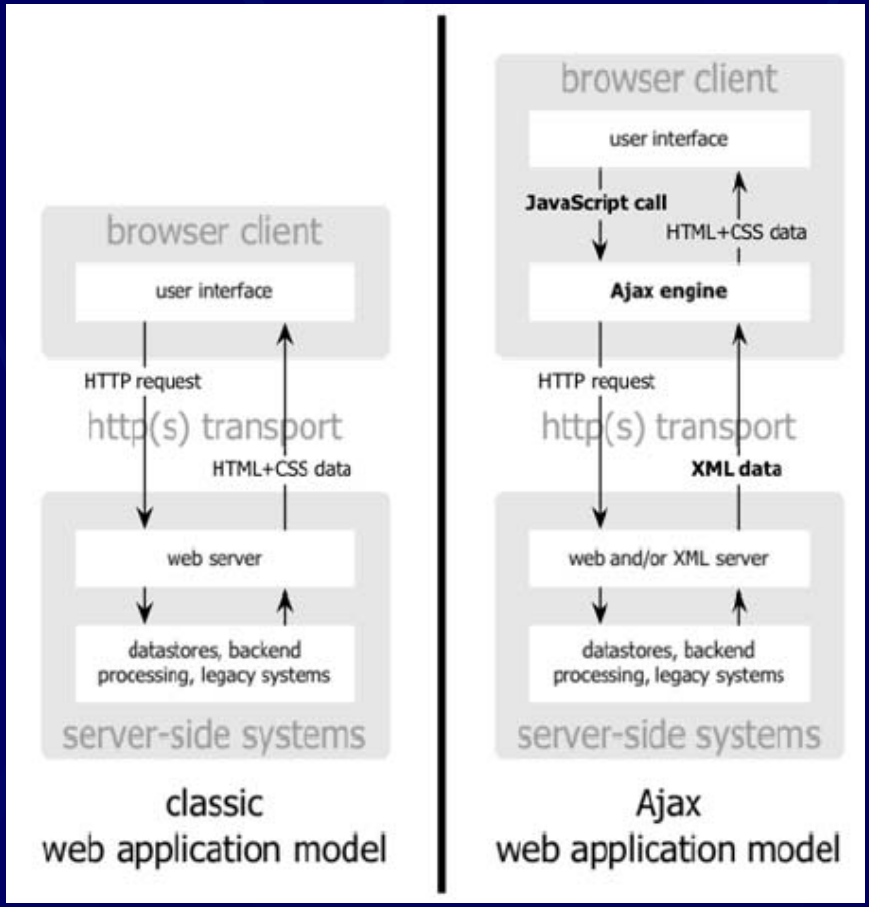


Partial UI Updates



Classic Vs. Ajax Web

No more single request / response restrictions



Potential Problems

- Breaks back button support
- URL's don't change as state changes
- SEO / Search Engine Indexing
- Cross Browser Issues can be a pain
- Cross-domain requests (SOA, WS)
- Viewable Source Code
- Client side business logic
- Server Load if not properly coded
- Concurrency

Security Concerns (I)

- JavaScript applications are easily decoded and reengineered
- On-demand (server side) loading will not help if you load your entire application and business logic in the client's memory
- Obfuscation makes it more difficult, but can also generate bugs

Security Concerns (II)

- XMLHttpRequest is nothing more than a normal form submission
  - Authentication elements
  - Session cookies
  - Blank Referrer by default ← <u>You should manually set this header in your API</u>
- Get / Post Verbs are sent in plain text
  - Consider encrypting requests and obfuscating responses
- Developers forget to send sensitive data over SSL

Ajax Request Header

Live HTTP headers

Headers | Generator | Config | About

HTTP Headers

http://www.robgonda.com/blog/projects/ajaxcfc/examples/echo/echoTest.cfc

POST /blog/projects/ajaxcfc/examples/echo/echoTest.cfc HTTP/1.1
Host: www.robgonda.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.0.1) Gecko/20060111 Firefox/1.5.0.1
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q...
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Referer: http://www.robgonda.com/blog/projects/ajaxcfc/examples/echo/
Content-Length: 235
Cookie: __utmz=235058050.1137544973.167.4.utmccn=(organic)|utmcsr=google|utmctr=ajaxcfc|utmcmd...
Pragma: no-cache
Cache-Control: no-cache
method=init&callCount=1&c0-scriptName=null&c0-methodName=echo&c0-id=7957_1142095689625&c...

Ajax Request Header

- Treat it just like a regular request
- Validate:
  - User-Agent (IE, Mozilla, Gecko)
  - Referrer: make sure it was posted from your site
  - Cookie: validate valid session
  - Verbs: require post vs. get

Security Rules

- Don't trust user input
- Do not trust client side validation
- Do not trust server side AJAX validation
  - Will improve user experience
  - Will not reduce code, only increase it.
  - You still need to re-validate in the server side
- Do not assume every Ajax request is real
- **Keep you business logic in the server**

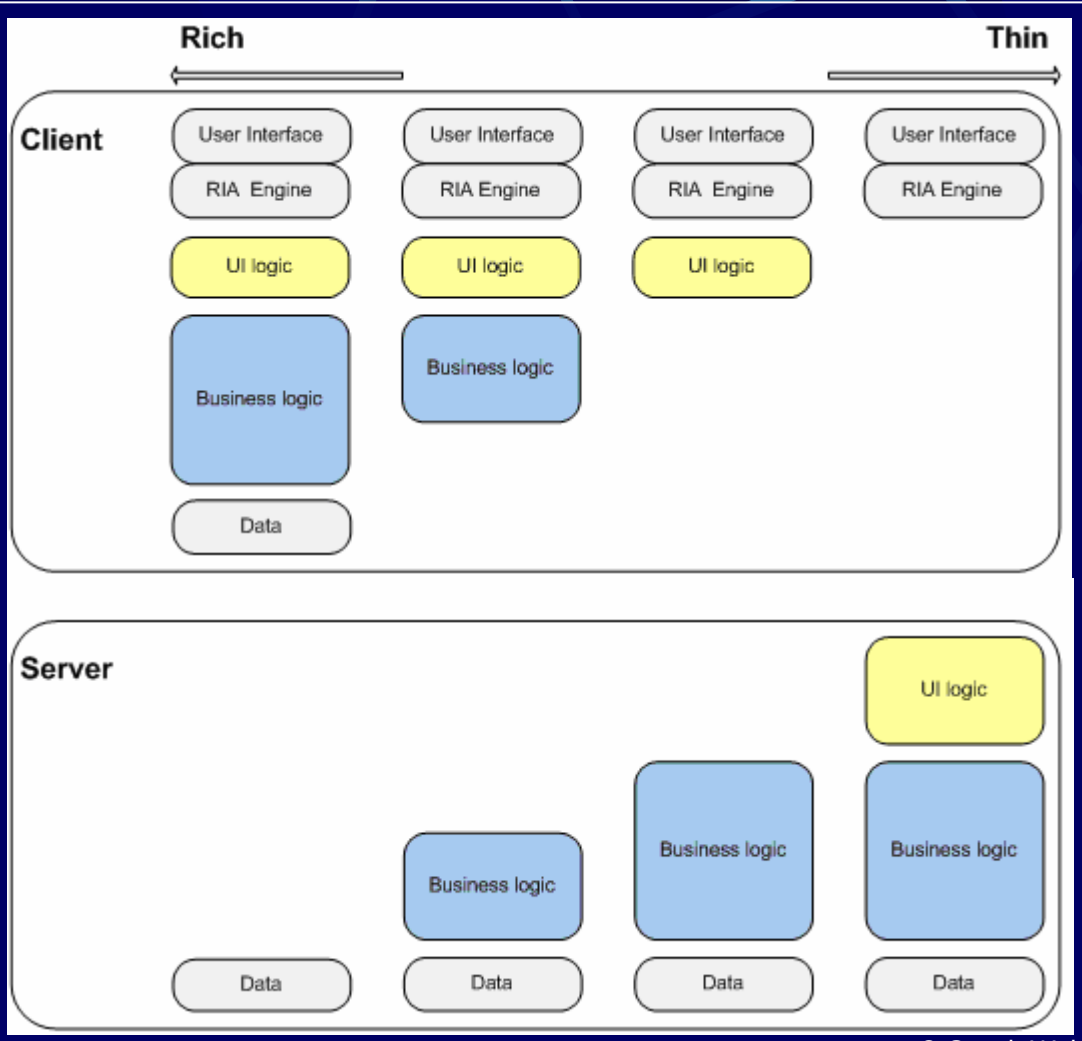
Do not expose business logic

- Most important aspect for Enterprise Applications
- Ajax uses JavaScript, but it does not have to reside in the client
- Use remote calls only as a transport layer
- Transport state and instructions, not raw data
- MVC
  - Model must remain on the sever
  - View or presentation layer is managed with DOM
  - Controller layer simply requests commands and dynamically evaluates them

What to transport

- Data
  - XML
  - WDDX
  - SOAP
  - JSON (JavaScript Object Notation)
  - JavaScript Native Objects (one way)
- Instructions
  - JavaScript instructions to be dynamically evaluated using eval()

Rich / Thin Client



© Coach Wei

The Magic Functions

- Traditionally you would use DOM
  - Cross browser hell
  - createNode is a pain
- innerHTML
  - For modifying content
- eval()
  - Dynamic expression evaluation

The eval() function

- Dynamic evaluation
  - ONLY in the client
  - NEVER in the server
    - Evaluation in the server could cause xml or sql injection.
- Evaluate innerHTML instruction
  - i.e. result = “ $(‘div1’).innerHTML = ‘text’ ”;
  - eval(result);

Weight and Performance

- Minification Vs. Obfuscation
  - A minifier removes the comments and unnecessary white space from a program.
  - An obfuscator also minifies, but it will also make modifications to the program
  - Any transformation carries the risk of introducing a bug.
  - Easier to minify dynamically generated code than obfuscate it
- GZIP
  - 15% of original file. Makes Minication and Obfuscation file size modification almost irrelevant.

Debugging

- window.onerror = function(message, url, lineNum) {
  alert("Error: '" + message + "'. At " + url + ", Line " + lineNum);
  }
- Use onerror to trigger a simple Ajax call that never ever changes to log the error in the server side (text file, xml, database)
- HTTP Traffic Sniffers
  - Live HTTP Headers (Firefox)
  - Firebug (Firefox) ← Rob likes this
  - Fiddler (IE, Firefox with some adjustments)
  - Ethereal (Advanced, multipurpose)
  - ServiceCapture (HTTP, AMF, JSON)

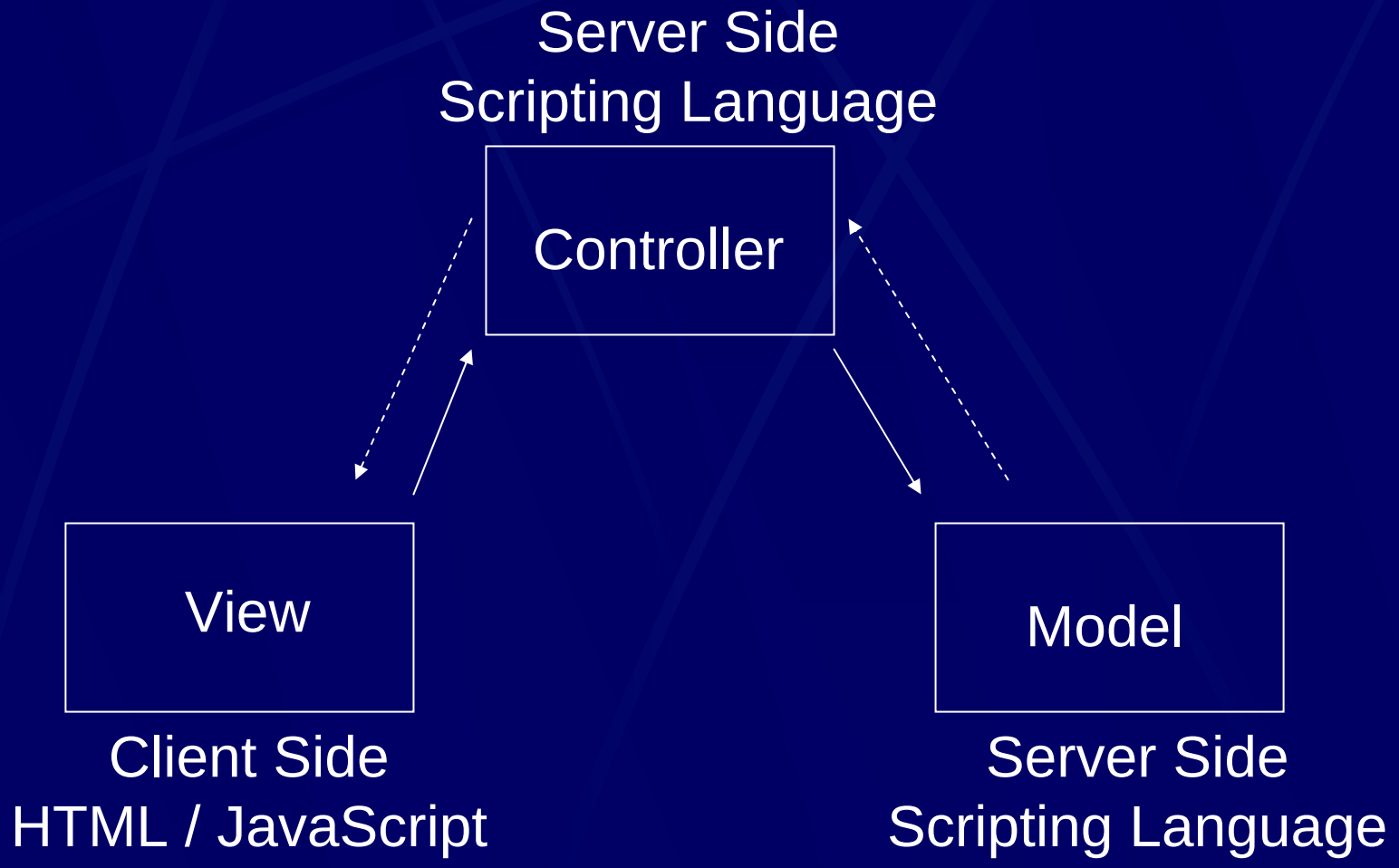
Cross-site restrictions

- Impediment to communicate with other domains is good. Cross-domain requests are a security hole
- There is no real need for it, Web Services should be consumed by server side scripting
- If really needed, use iframes technique instead

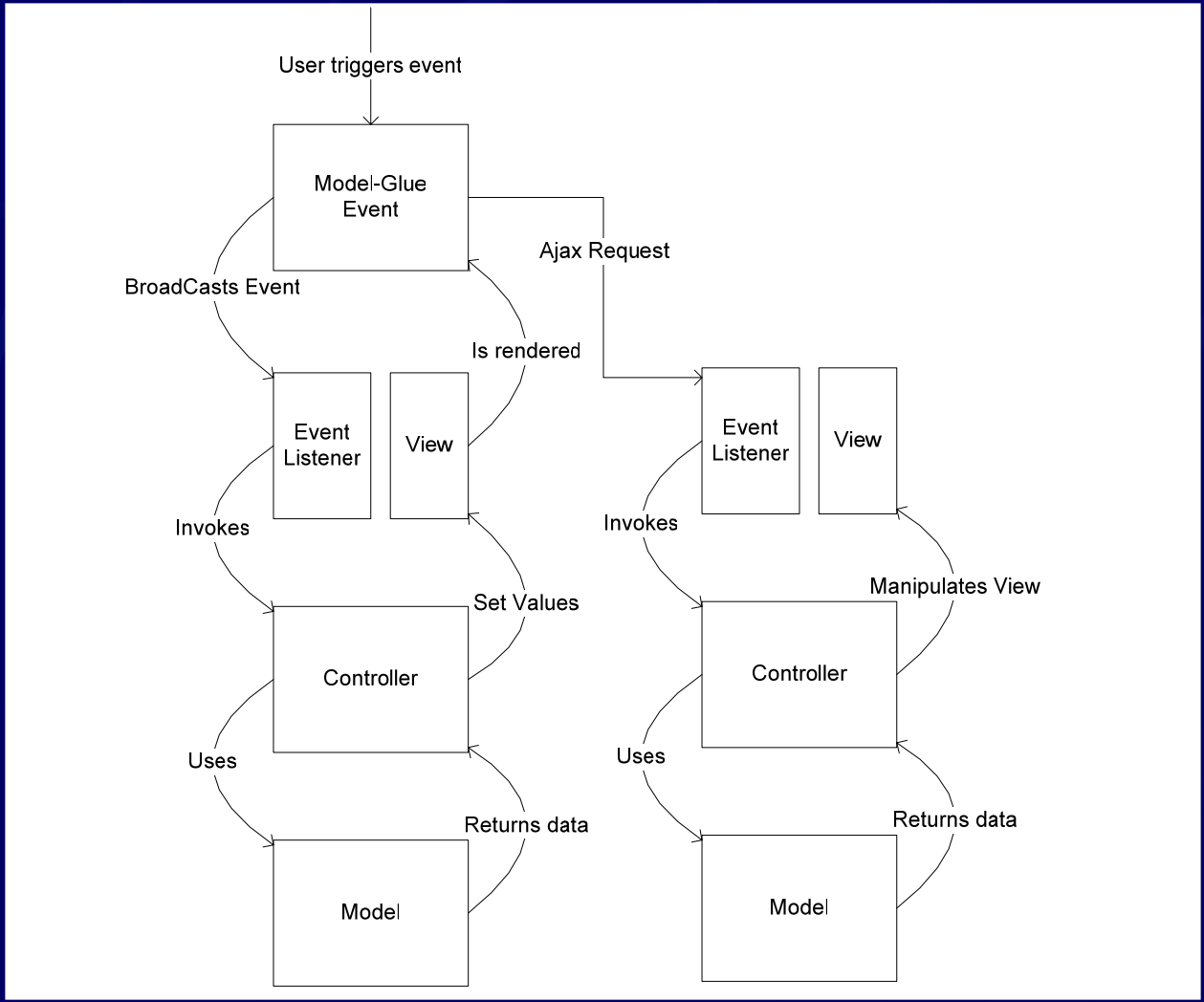
MVC

- Model-view-controller (MVC) is a software architecture that separates an application's data model, user interface, and control logic into three distinct components.
- Clear distinction between the presentation layer and business logic

MVC Diagram


Server Side
Scripting Language
Controller
View
Model
Client Side
HTML / JavaScript
Server Side
Scripting Language

MVC Pattern Data Flow



AJAX Libraries

- Many people opt for AJAX libraries.
- Provides many advantages
  - Development time
  - Sync / Async
  - Serialization
  - Multithread / batches
  - Error handling
  - Logging
  - Security, encryption, obfuscation
- Disadvantages : NONE!
  - Weight?

JavaScript AJAX Libraries

- Libraries
  - Prototype (the most popular, used by RoR)
  - Dojo (used by Open Ajax, XAP, Kabuki)
  - DWR (integrate with Java Struts and JSF)
- Toolkits
  - Microsoft Altas (commercial)
  - Open Ajax (IBM, Zimbra, Dojo | open source)
  - Open Laszlo (Flash and Ajax)
  - Tibco GI
  - Backbase

ColdFusion Frameworks

- Fusebox (Steve Nelson and Gabe Roffman)
- Mach II (Ben Edwards and Hal Helms)
- Model Glue (Joe Rinehart)
- SAM (Simon Horwith)
- TheHUB (Neil Ross)

ColdFusion AJAX Libraries

- ajaxCFC
  - OOP, CF extends objects, Design Patterns
  - Built-in error handling, security, debugging
  - Capability to integrate with Model Glue, Mach II
- CFAjax
  - First Ajax CF Framework
- JSMX
  - Client side library only
- Simple Remote Scripting (SRS)
  - Uses iframes

Model Glue

- An Implicit Invocation framework simplifying use of the Model View Controller design pattern in ColdFusion applications.
- A framework encouraging clear separation of Model, View, and Controller
- Written by Joe Rinehart
- Released under the Lesser GPL, meaning that it's free to download, use, and alter.

ajaxCFC

- AjaxCFC is a Coldfusion/JavaScript framework meant to speed up ajax application development and deployment by providing developers seamless integration between JavaScript and ColdFusion, and providing built-in functions, such as security and debugging, to quickly adapt to any type of environment and helping to overcome cross-browser compatibility problems.
- Programming with ajaxCFC involves extending components and creating your own ajax façades.
- Intergration with Model-Glue, one of the most popular MVC frameworks widely used by the ColdFusion community.
- Works with ColdFusion MX 6.0, 6,1, 7.0 and Blue Dragon.
- Automatically handles complex object transmitted from the client to the server and vice versa.
- Licensed under the Apache License Version 2.0, by Rob Gonda.

Extending Ajax

- Flash plays friendly with Ajax
- The Flash Platform does not have the browser limitations
- Dojo uses Flash for persistent storage
- Flash 8’s IO classes have been used for file transfers
- Rob uses Flash for XML Sockets messaging services

Switch to Live Demos

- Simple Echo / Traditional Ajax
- Model Glue Echo
- Model Glue Ajax Contact Manager
- Model Glue Ajax / XMLSockets Contact Manager
- MVC AJAX RSS Reader

Thank You

- Questions / Comments?
- Blog: http://www.robgonda.com
- Corp: http://www.ichameleongroup.com
- email: rob@robgonda.com