

Creating and Consuming Web Services with CFML

Charlie Arehart
charlie@carehart.org



CFUNITED – The premier ColdFusion conference www.cfunitied.com

Topics

- Web Service Basics
- Publishing Your First Web Service
 - ✓ Building, testing your web service
- Consuming the Web Service in CFML
 - ✓ Several ways to invoke them, pass data
- Web Service Details and Caveats
- Where to Learn More



June 28th – July 1st 2006

About Your Speaker

- CTO Garrison Enterprises since Apr 2006
 - ✓ Formerly CTO, New Atlanta (BlueDragon)
- 9 yrs CF experience (24 yrs in Enterprise IT)
- Co-author, ColdFusion MX Bible (Wiley)
 - ✓ Frequent contrib. to ColdFusion Dev Journal
- Past accomplishments of note
 - ✓ Tech Editor, CFDJ
 - ✓ Certified Adv CF Developer (4, 5, MX), Certified Instructor, Team Macromedia Member
- Frequent speaker: UGs, conferences worldwide



June 28th – July 1st 2006

What's a Web Service?

- Simple terms: a web page designed to be consumed by software, not by humans
 - ✓ Who remembers WDDX?
 - Could “serialize” data to Allaire-specified XML
 - Could output from page request, pass to caller
 - ✓ Problems:
 - Non-standard (though see openwddx.org)
 - Wasn't a function call/response approach
 - As used in similar service approaches: RPC, CORBA



June 28th – July 1st 2006

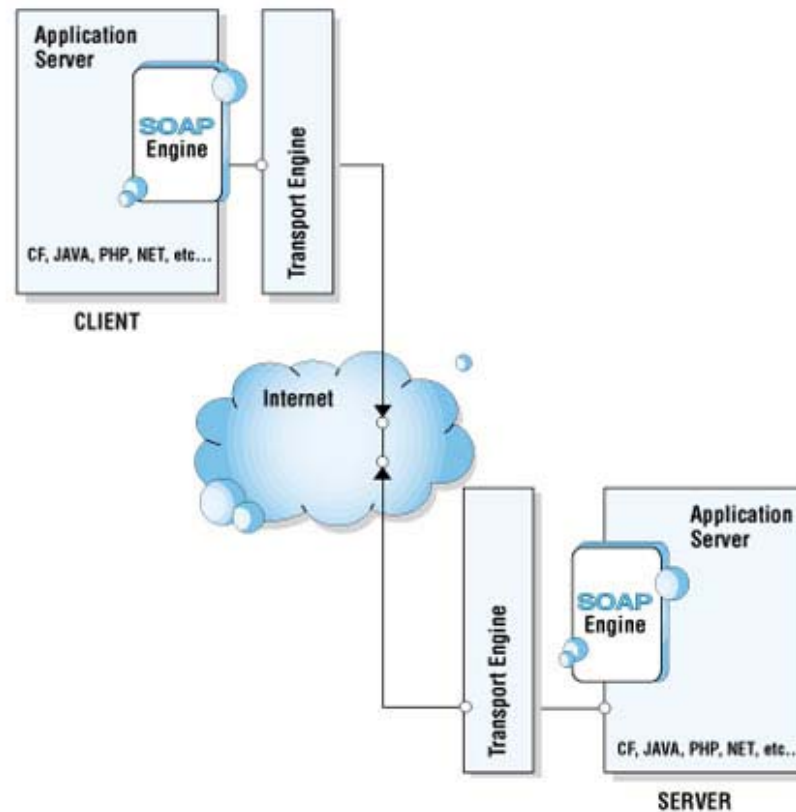
What's a Web Service?

- One technical definition of web services:
 - ✓ Remotely consumable software component published on the web through a universal interface
- IT world moving to service-oriented architecture
 - ✓ One application provides services to another



June 28th – July 1st 2006

Web Service Communications



Coalescing Standards

- Standards have come together for web services
 - ✓ SOAP – Simple Object Access Protocol
 - ✓ WSDL – Web Services Description Language
 - ✓ UDDI – Universal Description, Discovery, and Integration
 - ✓ XML – eXtensible Markup Language
- Good news: CFML hides need for you to understand any of these, for the most part



June 28th – July 1st 2006

Some Examples

- Bottom line: web services enable an organization to expose their data to other applications
 - ✓ Perhaps running on other platforms within an org.
 - ✓ Or applications running in other organizations
 - ✓ Sometimes referred to as *syndication* of content
- Possible private implementations
 - ✓ Supplier exposing inventory status to partners
 - ✓ Vendor exposing products to affiliates
 - ✓ Government organization sharing data with other orgs
 - ✓ Obtain weather, stock quotes for your web app without screen scraping
 - ✓ To name a few

June 28th – July 1st 2006



Some Examples

- Commercial examples

- ✓ Amazon, Google, USPS, UPS, FedEx



- Available service listings

- ✓ <http://Xmethods.net>
- ✓ <http://www.serviceobjects.com/products/default.asp>



June 28th – July 1st 2006

CFML Makes It Easy

- Many languages support web services, including use of those protocols, creation of XML, etc.
 - ✓ Including Java, ASP.NET, PERL, etc.
 - ✓ These often use APIs involving lots of code



June 28th – July 1st 2006

CFML Makes It Easy

- CFMX and BlueDragon both support very easy publication and consumption of web services
 - ✓ **Don't** need to know Java, nor SOAP, WSDL, UDDI
 - ✓ *Don't even need to understand XML*
 - ✓ Web Service processing in CFML is very easy
- Other end of conversation need not be CFML
 - ✓ Web Services are designed to be agnostic as to language and platform
 - ✓ A CF web service could be consumed by a Java app, or .NET, etc.
 - *And vice versa*
- Note: if you're doing CFCs already, you don't need to change them to using web services
 - ✓ Unless you're interested in exposing data to non-CF apps



June 28th – July 1st 2006

Publish/Consume

- **Publication** of web svc in CFML is simple
 - ✓ Simply expose a CFC method as “remote”
 - ✓ Return data to caller using CFRETURN
- **Consumption** is equally easy
 - ✓ Use CFINVOKE (or CFOBJECT/createObject)
 - ✓ Invoke methods of the exposed service
- We'll explore each of these in remainder of talk



June 28th – July 1st 2006

Publishing Your First Web Service

- Assume we have some data to publish
- Can publish any kind of data via web svc
 - ✓ Simple string
 - ✓ Array
 - ✓ Structure
 - ✓ Array of structures
 - ✓ CFML query result set (with a caveat, discussed later)
 - ✓ XML object (using CFMX and BlueDragon's support of XML)
 - ✓ To name a few



June 28th – July 1st 2006

Publishing Your First Web Service

- Need merely create a CFC function that returns that data, expose function as “remote”
 - ✓ If you’re new to CFCs’, they have many other benefits and features
 - ✓ Their use for web services is actually one of their easier purposes to understand



June 28th – July 1st 2006

Building the Component Functions

- Let's build Hello World web service

```
<!-- hello.cfc -->
<CFCOMPONENT>
  <CFFUNCTION NAME="GetHello" ACCESS="REMOTE"
    RETURNTYPE="String">
    <CFRETURN "Hello World">
  </CFFUNCTION>
</CFCOMPONENT>
```

- That's really all there is to it!
- Now any web service client (caller) can invoke this web service (hello.cfc) and its method (GetHello)
 - ✓ First time it's called, CFMX/BlueDragon will generate WSDL needed for callers



June 28th – July 1st 2006

Calling Your Web Service

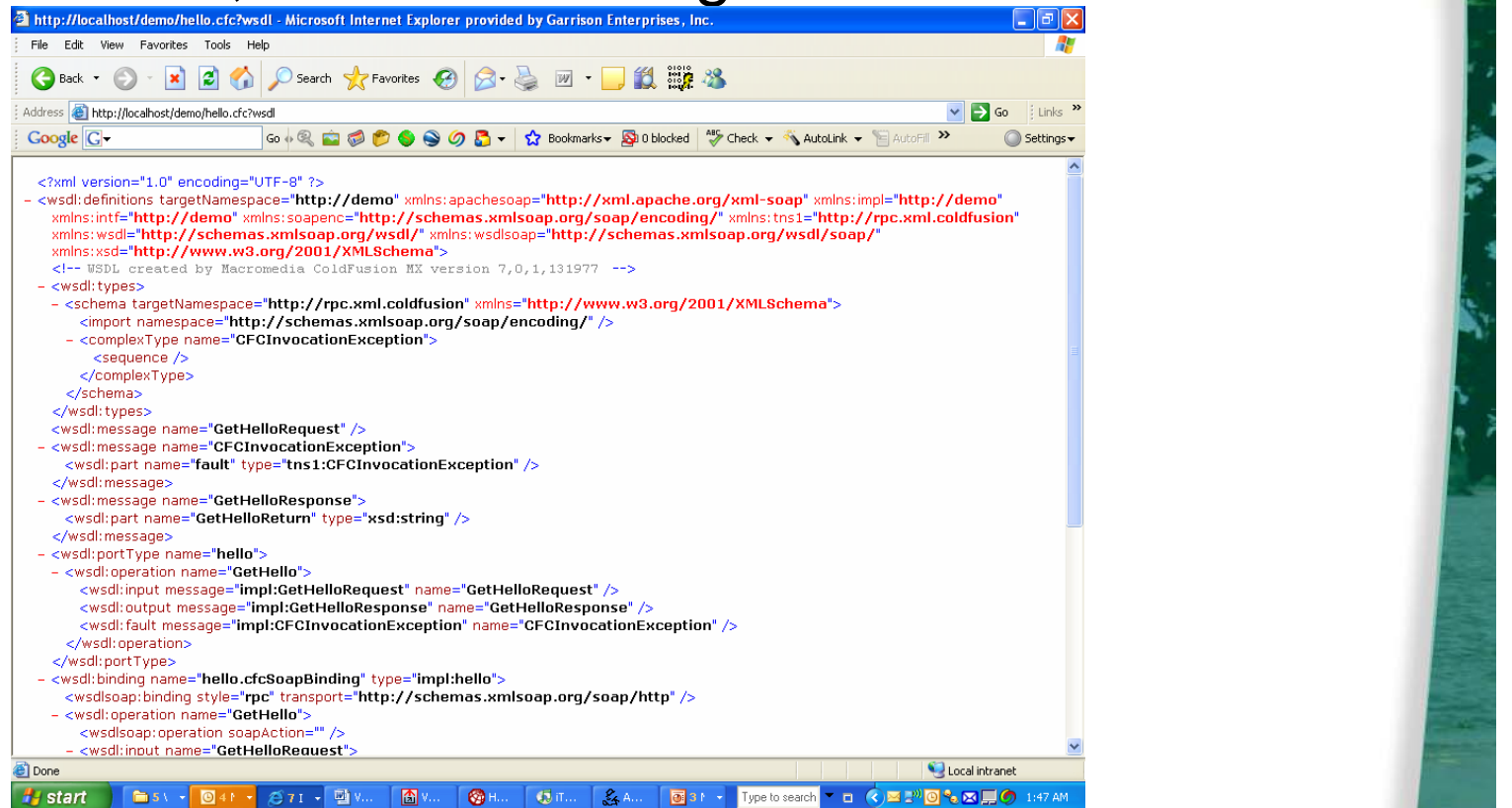
- Can call from any language/platform that supports web services
 - ✓ Simply need to refer to:
 1. Domain name (and port, if needed) that's used to access code on CFMX/BlueDragon
 2. Directory in which CFC is located
 3. CFC name
 4. ?WSDL indicator
 - <http://localhost/demo/hello.cfc?wsdl>
 - ✓ Since I stored the CFC in my demo directory



June 28th – July 1st 2006

Viewing WSDL

- For testing, can enter the web service URL in your browser, to see resulting WSDL



```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://demo" xmlns:apachesoap="http://xml.apache.org/xml-soap" xmlns:impl="http://demo"
  xmlns:intf="http://demo" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns1="http://rpc.xml.coldfusion"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsoap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <!-- WSDL created by Macromedia ColdFusion MX version 7,0,1,131977 -->
  - <wsdl:types>
  - <schema targetNamespace="http://rpc.xml.coldfusion" xmlns="http://www.w3.org/2001/XMLSchema">
    <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
    - <complexType name="CFCInvocationException">
      <sequence />
    </complexType>
  </schema>
  </wsdl:types>
  <wsdl:message name="GetHelloRequest" />
  - <wsdl:message name="CFCInvocationException">
    <wsdl:part name="fault" type="tns1:CFCInvocationException" />
  </wsdl:message>
  - <wsdl:message name="GetHelloResponse">
    <wsdl:part name="GetHelloReturn" type="xsd:string" />
  </wsdl:message>
  - <wsdl:portType name="hello">
  - <wsdl:operation name="GetHello">
    <wsdl:input message="impl:GetHelloRequest" name="GetHelloRequest" />
    <wsdl:output message="impl:GetHelloResponse" name="GetHelloResponse" />
    <wsdl:fault message="impl:CFCInvocationException" name="CFCInvocationException" />
  </wsdl:operation>
  </wsdl:portType>
  - <wsdl:binding name="hello.cfcSoapBinding" type="impl:hello">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
  - <wsdl:operation name="GetHello">
    <wsdlsoap:operation soapAction="" />
  - <wsdl:input name="GetHelloRequest">
```



June 28th – July 1st 2006

Consuming via CFINVOKE

- To use the web service within CFML, use either CFINVOKE, CFOBJECT, or createObject
- CFINVOKE:
 - ✓ Calls web service (using URL described earlier, with ?wsdl as querystring), names method to execute, and variable to hold returned result

```
<cfinvoke  
  webservice="http://localhost/demo/hello.cfc?wsdl"  
  returnvariable="fromhello" method="GetHello">
```

```
<cfoutput>#fromhello#</cfoutput>
```



Consuming via CFOBJECT/CreateObject

- Can also use CFOBJECT/CreateObject instead
 - ✓ Slight difference from CFINVOKE
 - They return an object representing the web service
 - Then can invoke method in CFML as with other objects

```
<cfobject webservice="http://localhost/demo/hello.cfc?wsdl" name="fromhello">  
<cfoutput>#fromhello.GetHello()#</cfoutput>
```

Or

```
<cfscript>  
fromhello=createobject("webservice", "http://localhost/demo/hello.cfc?wsdl");  
writeoutput(fromhello.GetHello());  
</cfscript>
```



June 28th – July 1st 2006

Calling a “Real” Web Service

- Invoking xmethods.net Temperature svc
 - ✓ Reports current temperature for given zipcode
 - ✓ Available methods/properties docs at site

```
<cfinvoke
```

```
  webservice="http://www.xmethods.net/sd/2001/DemoTemperatureService.wsdl" method="GetTemp"  
  returnvariable="weather">
```

```
<cfinvokeargument name="zipcode" value="30005"/>
```

```
</cfinvoke>
```

```
<cfdump var="#weather#">
```



June 28th – July 1st 2006

Passing Input to Web Service

- Simple examples so far took no input
- Let's change to say 'hello' to us personally
 - ✓ Can add a new CFFUNCTION to existing CFC
 - multiple functions with different names

```
<CFFUNCTION NAME="GetPersonalHello"  
  ACCESS="REMOTE" RETURNTYPE="String">  
  <CFARGUMENT NAME="fname" TYPE="string">  
  <CFRETURN "Hello, #fname#">  
</CFFUNCTION>
```



Passing Input to Web Service

- Can pass named argument on CFINVOKE:

```
<cfinvoke webservice="http://localhost/demo/hello.cfc?wsdl"  
  returnvariable="fromhello" method="GetPersonalHello"  
  fname="charlie">
```
- Or, using CFINVOKEARGUMENT

```
<cfinvoke webservice="http://localhost/demo/hello.cfc?wsdl"  
  returnvariable="fromhello" method="GetPersonalHello">  
  <cfinvokeargument name="fname" value="charlie">  
</cfinvoke>
```
- ✓ Useful to build args dynamically
 - such as CFIF, CFLOOP inside the CFINVOKE



Testing Web Services

- Can invoke methods on browser request:
`http://localhost/demo/hello.cfc?wsdl&method=GetHello`
- Can even pass simple string as args on URL in a browser request:
`http://localhost/demo/hello.cfc?wsdl&method=GetPersonalHello&fname=charlie`

**Mistaken URL
in notes**



Web Service Details/Caveats

- Exception Handling
 - ✓ Web service requests may fail
 - ✓ Consider cftry/cfcatch to detect/handle errors
- Timeout
 - ✓ CFMX 6.1 added ability to timeout web service requests
 - how long you're willing to wait for a reply



June 28th – July 1st 2006

Web Service Details/Caveats

■ Security

- ✓ Can secure CFC using either web server authentication
 - just as you can limit access to any web page
 - CFINVOKE offers USERNAME/PASSWORD
- ✓ Could even implement your own alternative attributes for authentication, and test for that in your CFC method
- ✓ Can secure in CFML using ROLE attribute on CFFUNCTION
 - Tied to CFLOGIN/CFLOGINUSER tags
 - See CFMX documentation for more details



June 28th – July 1st 2006

Web Service Details/Caveats

- Caching Web Service object
 - ✓ As we might cache a query resultset if it doesn't change often, can do with web svc
 - ✓ No current feature to cache web service results
 - Can do it yourself, storing result in shared scopes (session/application/server)
 - Use some timing mechanism to determine when to refresh result, re-execute web service invocation



June 28th – July 1st 2006

Web Service Details/Caveats

- Beware: non-CF consumers won't understand if you return CF query result
 - ✓ Can instead convert into an array of structures
 - ✓ Consider following UDFs at the cflib.org site
 - QueryToArrayOfStructures: <http://cflib.org/udf.cfm?ID=10>
 - ArrayOfStructuresToQuery: <http://cflib.org/udf.cfm?ID=287>

 - QueryToStructOfArrays: <http://cflib.org/udf.cfm?ID=470>
 - QueryToStructOfStructures: <http://cflib.org/udf.cfm?ID=523>



June 28th – July 1st 2006

Web Service Details/Caveats

- How can you know the type of data returned from web svc/passed into CFC?
 - ✓ It it a query? an array? a structure? something else?
 - ✓ You may need to know its type to determine how to process it. See “typeof”, at:
 - <http://cflib.org/udf.cfm?ID=689>
 - Reports if something is a array, struct, query, string, date, numeric, boolean, binary, wddx, xml object, or even a custom function (udf)



Why Web Services Will Succeed

- Why will web services succeed?
 - ✓ They're relatively simple
 - ✓ People generally agree on their use
 - ✓ Major IT vendors and svc supporting them
 - ✓ People are excited about them, using them
- For CFers, best thing is they're so easy
 - ✓ You can easily introduce them into your org



June 28th – July 1st 2006

More You Can Learn

- CFMX docs elaborate on many additional topics
 - ✓ Working with WSDL files
 - ✓ Consuming web svcs not generated by CFMX
 - ✓ Calling web services from a Flash client
 - ✓ Catching errors when consuming web services
 - ✓ Configuring web svcs in CFMX Administrator
 - ✓ Conversions between CF/WSDL datatypes
 - ✓ Defining data types for web services
 - ✓ Handling complex data types
 - ✓ Integrating with Dreamweaver MX



June 28th – July 1st 2006

More You Can Learn

- Available “tips & tricks” presentation to show some of these and more
 - ✓ Offered Friday here at CFUnited
- Articles that follow offer pointers to finding publicly available web services that you can explore



June 28th – July 1st 2006

Learning More

- Macromedia Documentation
 - ✓ 6.1: *Developing ColdFusion MX Applications*, Chapter 32
 - ✓ 7: *ColdFusion MX Developer's Guide*, Chapter 36
 - ✓ Available at livedocs.macromedia.com
- Books
 - ✓ CFMX Bible, Wiley (Churvis, Helms, Arehart), Chapter 25
 - ✓ Programming ColdFusion MX (Brooks-Bilson), Chapter 25
 - <http://www.webreference.com/programming/coldfusion/1/index.html>
 - ✓ And others



June 28th – July 1st 2006

Learning More

- CFDJ Web Services Articles

- ✓ Jul 02, “CFMX & Web Services”, Ron West
- ✓ Jul 02, “A Quick and Easy Web Service with ColdFusion MX”, Kevin Schmidt
- ✓ Jul 02, “Using ColdFusion Components--Part 2”, Ben Forta
- ✓ Feb 03, “Consuming Amazon Web Services”, Charlie Arehart
- ✓ And others



June 28th – July 1st 2006

Other CFML/Web Svc Resources

- “A Beginner's Guide to Creating and Consuming Web Services with ColdFusion and Flash”
 - ✓ http://assets.macromedia.com/devnet/coldfusion/articles/beginner_ws_print.html
- “How May I Be of Web Service?”
 - ✓ <http://cfdj.sys-con.com/read/41554.htm>
- “CFMX & Web Services”
 - ✓ <http://cfdj.sys-con.com/read/41624.htm>
- “Creating a Web Service in ColdFusion MX”
 - ✓ <http://www.macromedia.com/desdev/mx/coldfusion/articles/webservices.pdf>



June 28th – July 1st 2006

Other CFML/Web Svc Resources

- “Macromedia MX: Components and Web Services” , Jeremy Allaire
 - ✓ http://www.macromedia.com/desdev/mx/coldfusion/whitepapers/components_ws.pdf
- “Creating ColdFusion web service pages with Macromedia Dreamweaver MX”
 - ✓ http://www.macromedia.com/support/dreamweaver/content/websrv_cf/
- “Creating a Web Service Application Using ColdFusion MX and Dreamweaver MX”
 - ✓ <http://www.macromedia.com/desdev/mx/dreamweaver/articles/webservice.html>
- “Crossing the .NET Divide: CFMX, Web Services, and .NET”
 - ✓ <http://cfdj.sys-con.com/read/47199.htm>



June 28th – July 1st 2006

Other CFML/Web Svc Resources

- Service-specific, but perhaps dated
- “Consuming Amazon.com web services with CFMX ”
 - ✓ <http://www.macromedia.com/desdev/mx/coldfusion/articles/wsamazon.html>
- “Techniques for Invoking the Google Web APIs service ”, Kevin Hoyt
 - ✓ <http://www.macromedia.com/desdev/mx/coldfusion/articles/googlews.html>



June 28th – July 1st 2006

Summary

- Web Services open new possibilities for CFML developers
- Very easy to both create and consume in CFML
 - ✓ CFC method w/ “Access=Remote” publishes
 - ✓ CFINVOKE/CFOBJECT/CreateObject consume
- Don't need to understand SOAP, WSDL, nor XML (usually)



June 28th – July 1st 2006

Summary

- Allows you to expose your CFML based processing to non-CFML clients
- CFMX and BlueDragon support web svcs
- CFML developers can lead the charge to using web services in their orgs!
- Questions: charlie@carehart.org



June 28th – July 1st 2006