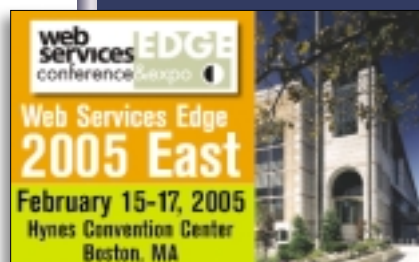


COLD FUSION Developer's Journal

ColdFusionJournal.com

October 2004 Volume: 6 Issue: 10



See page 43 for details

Editorial

Getting Ready for MAX

Simon Horwith page 5

Community Focus

'Tis the Season...

Simon Horwith page 7

CF101

Creating and Using User-Defined Functions

Jeffrey Houser page 28

CFUGs

ColdFusion User Groups

page 46

FOUNDATIONS

Extreme Programming

Hal Helms page 48

RETAILERS PLEASE DISPLAY
UNTIL DECEMBER 31, 2004

\$9.99US \$9.99CAN



SYS-CON
MEDIA



Data: Data Classification Using a Digital Taxonomy *As the volume of digital content grows, so does the need for efficient and accurate data retrieval*

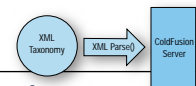
David Athey



12

Programming: Three Steps to Better Leadership *Start with yourself*

Reuben Poon



16

Web Apps: Controlling Page Sequencing with an Event-Driven State Machine *Write more robust, secure, and reusable code*

David Chandler



22

Best Practices: CFC Data Exchange *Open up the doors to a whole new realm of development solutions using CFML*

Simon Horwith



32

Tips & Tricks: HTTP Status Codes: Do the Unthinkable *Implement threading using CFML*

Brandon Harper



36

OLAP: Using OLAP with ColdFusion *Take data analysis to the next level using MS Analysis Services*

Mark Murphy



40

macromedia®





This is:

☐ Video

☐ Flash

☐ Don't bother me. I'm staring.

The line between Flash and Video is forever gone.

Travel with us to Scandinavia for a behind the scenes tour
of the award-winning, genre-bending Volvo V50 website.

macromedia.com/go/volvo



THE GRASS REALLY IS GREENER ON THE ***SERVERSIDE.***

SUPERIOR MANAGED HOSTING

- Intelligent Routing
- Redundancy
- Network Security
- Service Level Agreement
- Environmental Controls
- Network Uptime Guarantee
- Scalability
- OC3/SONET Backbone
- Backup Power
- Physical Security
- Fire Protection
- Server Hardware Guarantee

IF YOU'VE BEEN SEARCHING for a reliable managed hosting partner, your search for greener pastures is over. There is no longer a need to settle for inferior support and lack of accountability. ServerSide takes the guess work, and the associated hassles, out of working with a technology partner.

ServerSide provides managed web hosting solutions for a wide range of customers including; Fortune 500 corporations, small and medium sized businesses and non-profit organizations located across the United States and around the World.

We provide a single point of accountability for all your web hosting infrastructure needs — while adding value, not cost.

>> SPECIAL OFFER

Sign up online for a new shared hosting account now at www.serverside.net and ServerSide will waive the set-up fee.

ENTER CODE: mxd2004

The grass is greener at
www.serverside.net
888.682.2544
hosting@serverside.net



Jeremy Allaire, *founder emeritus, macromedia, inc.*
Charlie Arehart, *CTO, new atlanta communications*
Michael Dinowitz, *house of fusion, fusion authority*
Steve Drucker, *CEO, fig leaf software*
Ben Forta, *products, macromedia*
Hal Helms, *training, team macromedia*
Kevin Lynch, *chief software architect, macromedia*
Karl Moss, *principal software developer, macromedia*
Michael Smith, *president, teratech*
Bruce Van Horn, *president, netsite dynamics, LLC*

editorial

editor-in-chief

Simon Horwith simon@sys-con.com

technical editor

Raymond Camden raymond@sys-con.com

executive editor

Jamie Matusow jamie@sys-con.com

editor

Nancy Valentine nancy@sys-con.com

associate editors

Gail Schultz gail@sys-con.com

Natalie Charters natalie@sys-con.com

research editor

Bahadir Karuv, PhD bahadir@sys-con.com

production

production consultant

Jim Morgan jim@sys-con.com

lead designer

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com

Richard Silverberg richards@sys-con.com

Tami Beatty tami@sys-con.com

Andrea Boden andrea@sys-con.com

contributors to this issue

David Athey, Dennis Baldwin, David Chandler,
Brandon Harper, Simon Horwith, Hal Helms,
Jeffrey Houser, Mark Murphy, Reuben Poon

editorial offices

SYS-CON MEDIA

135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

TELEPHONE: 201 802-3000 FAX: 201 782-9638

COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)

is published monthly (12 times a year)

by **SYS-CON Publications, Inc.**

postmaster: send address changes to:

COLDFUSION DEVELOPER'S JOURNAL

SYS-CON MEDIA

135 Chestnut Ridge Rd., Montvale, NJ 07645

©copyright

Copyright © 2004 by SYS-CON Publications, Inc.
All rights reserved. No part of this publication may
be reproduced or transmitted in any form or by any means,
electronic or mechanical, including photocopy
or any information, storage and retrieval system,
without written permission.

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

For promotional reprints, contact reprint
coordinator Kristin Kuhnle, kristin@sys-con.com.
SYS-CON Publications, Inc., reserves the right to
revise, republish and authorize its readers to use
the articles submitted for publication.

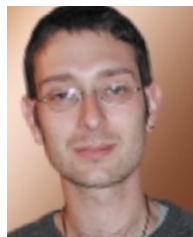
All brand and product names used on these pages
are trade names, service marks, or trademarks
of their respective companies.

Getting Ready for MAX

It's good to experiment – this holds true for any and every endeavor. Last month I tried something different with **CFDJ**, almost every article in the issue was about one topic – security. I've gotten a lot of feedback from readers and am happy to report that it was all very positive. Many of you wrote telling me that you really enjoyed seeing an issue devoted to one topic in-depth. I will do my best to organize issues that are focused on one issue in the future. If you have an idea(s) for focus areas you'd like to see **CFDJ** concentrate on, let me know. I have some ideas of my own, but I much prefer to base the magazine content on audience feedback. Next month's issue will also feature heavy focus on one topic: MAX conference coverage.

For those of you who aren't already familiar with MAX, this event is the largest conference of the year for users of Macromedia products (and is organized by Macromedia as well). The first was last year in Salt Lake City (prior to that, we had the annual DevCon Conference); this year the conference is in New Orleans from November 1–4. Next month we will feature more in-depth coverage of the conference than any prior year. Many regular contributors to **CFDJ** will be presenting at the conference, including Hal Helms, Ben Forta, Ray Camden, and myself. **CFDJ** has sent April Fleming, who you may remember reading about in Sean Corfield's CFUN 04 conference summary a few months ago, to cover the "MAX Experience." Historically, Macromedia (and Allaire) have always made big announcements about upcoming software at the conference, and several Blackstone topics were added to the session schedule at the last minute. I won't speculate on what that means, but obviously for the ColdFusion developers in attendance, Blackstone is going to be in everyone's thoughts. In addition, I will pen a post-conference **Community Focus** column that you won't want to miss. The only other thing I can tell you is that it will contain a lot of quotes.

Because last month's format was an experi-



By Simon Horwith

ment and I wanted to get feedback before using that format again, this month's issue doesn't focus on any one topic. My community column is about choosing sessions to attend at conferences and about a few of the MAX session descriptions on Macromedia's Web site that look interesting to me. Dennis Baldwin wrote an article about choosing between Flash Remoting and Web

services in Flash MX 2004. A couple of months ago I wrote a **Tales from the List** article about a trend in the increasing importance of metadata. Following up on that theme, David Athey has written a very interesting article in this month's issue about digital taxonomy. Another article I found interesting is Brandon Harper's article about using HTTP Status Codes in order to create asynchronous module calls as well as performing form submissions without an apparent page reload. Whether you like the technique or not, it's a good example of how you can creatively use the CFHTTP tag. If you've always wondered about (or never heard of) online analytical processing (OLAP), you'll definitely want to read Mark Murphy's article, which introduces the reader to the ideas behind data warehousing and how to use Microsoft's Analysis Services software.

— continued on page 31

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. He is a freelance software architect currently consulting with companies in London. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and consults on ColdFusion applications that leverage Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com
simon@horwith.com



Need a hand with your work? Get the MX Kollection.

Introducing MX Kollection - the suite of extensions designed to change the way you create dynamic web applications with Dreamweaver MX and MX 2004.

Our customers think of MX Kollection as the next level in Dreamweaver MX visual software development.

ColdFusion and PHP developers will find our product enabling them to visually develop e-Commerce, CMS, CRM, Backend and other web solutions with increased efficiency, quality and capability.

Create powerful dynamic lists

- Sort, filter and page result sets
- Perform multiple deletes
- Easily create Master/Detail lists

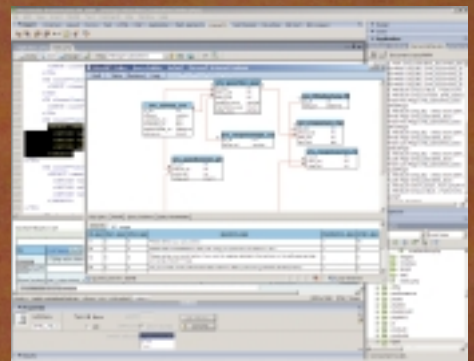
Build unified add/modify forms

- Client side and server side validation
- Insert into two tables
- Create form actions such as send mail or delete related record
- File/Image upload and resize

Create recordsets visually

- Build complex SQL queries across multiple tables quickly

... and many more



Interakt

macromedia
ALLIANCE PARTNER

Download the demo and see the features and benefits of our extensions at <http://www.interaktonline.com>

'Tis the Season...

...for CF conferences, that is

Several conferences have been officially announced, most notably:

- **MX Europe 2005** (www.mx europe.org): Formerly known as "CF-Europe" the conference now spans three days and includes Macromedia client and server technology session topics. It is the largest conference for Macromedia product users in Europe, and will be held in London at the end of January 2005.
- **Powered By Detroit** (www.poweredbydetroit.org/): This is the inaugural year for this two-day conference that focuses on ColdFusion/Flash integration. The conference is scheduled to be held in Detroit the beginning of April 2005.
- **CFUnderground VI** (www.cfconf.org/cf_underground6/): This is the sixth CFUnderground Conference, and as usual it promises to be a good way to start the conference season. Every year, Michael Smith and the folks at TeraTech host this fun and informal one-day conference on the eve of the "big CF conference." This year it's in New Orleans on Halloween!

All of the above conferences will feature star-studded lineups of speakers, and I strongly recommend attending them if you're able.

Of course, what really makes conference season is Macromedia's big annual conference. Every year, designers and developers from all over the world come together for MAX (www.macromedia.com/go/max/). MAX offers attendees and exhibitors the ability to learn from presenters who are the best in their field; the chance to have fun socializing and talking shop with each other and with leading experts in the industry; and the chance to see sneak-peeks of Macromedia products and initiatives yet to be released.

This year's speaker and topic lineups look to be the best yet, so for the benefit of the attendees at this year's MAX conference, I thought I'd offer some advice about choosing sessions to attend, as well as my thoughts on sessions and speakers not to be missed. I'm sure all of the presentations are going to be fantastic – my apologies to all of the presenters who, due to



By Simon Horwith

space limitations, are not listed here. If you were unable to attend MAX, this will serve as a guide for choosing your sessions at next year's conference and as a reminder of why you should do everything in your power to attend.

In selecting sessions at MAX (and most any conference) there are a few rules of thumb that I've found very good to follow (in order of priority):

1. There are certain speakers who, because of their presenting style and/or thoroughness, are worth seeing no matter what the subject matter of their session. Often but not always, these are well-known speakers.
2. Try to see Macromedia employees speak. These folks not only have expertise but also have inside information. It is not uncommon for them to discuss or show hidden gems in the products as well as provide insight about what the future may hold for a product.
3. If you don't know whether or not the speaker is a good presenter, read his or her bio. People who have had interesting careers or who hold interesting jobs are often good speakers. The other thing to look at is the presentation topic – sometimes a session topic and/or description is so interesting (or I have a practical interest in the topic) that I want to attend.

— continued on page 20

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. He is a freelance software architect currently consulting with companies in London. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and consults on ColdFusion applications that leverage Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com simon@horwith.com



Flash MX Web Services vs Remoting

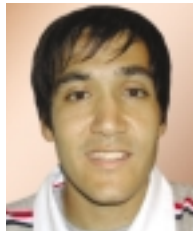
Which is right for your project?

Flash MX 2004 Pro Web service classes or Flash Remoting components for ActionScript 2.0: Which is right for you? This article offers some guidelines.

In just three years we've come a long way. It seems like an eternity since the days when I used to code CF pages with dynamic content served up to Flash, but let me stress how rudimentary it was. Flash back to December 14, 2001, when a little site called FlashCFM received Macromedia's Site of the Day designation. On several occasions I received e-mails asking me how on earth FlashCFM received this prestigious award. There was nothing spectacular about the site, no fancy graphics, Flash animation, or anything out of the ordinary. Yet the concept behind it was brilliant, or so I believed. As you can probably guess from its name, this site introduced the marriage of Flash and ColdFusion.

FlashCFM was a community site that educated developers about the power of Flash and ColdFusion integration. When I say that this process used to be rudimentary, let me give a quick example of ColdFusion code in a page called variables.cfm:

```
<cfoutput>&first_name=#query.first_name#&last_name=#query.last_name#</cfoutput>
```



By Dennis Baldwin

So the output would look like:

```
&first_name=Dennis&last_name=Baldwin
```

Flash would then retrieve this data and display it with a simple line of code:

```
loadVariables("variables.cfm", _root);
```

A very simple framework indeed (more like a CSV screenscraper or accessing a query string), but it proved to be the start of something wonderful. Flash forward to the present and we have such powerful technologies as Flash Remoting and Web service components. In this article I will go into detail about the pros and cons of each solution, as well as give you enough information to decide which is the best fit for your project. This will be demonstrated with a simple application that calls identical service methods, one using Flash Remoting and the other using the Web service classes. Enough history, let's get started!

Flash MX Pro Web Service Classes

If you've opened Flash MX 2004 Pro, then you're probably aware of the new Web service classes. Macromedia has done a great job in releasing components that allow Flash developers to easily consume Web services. Throughout the rest of this article

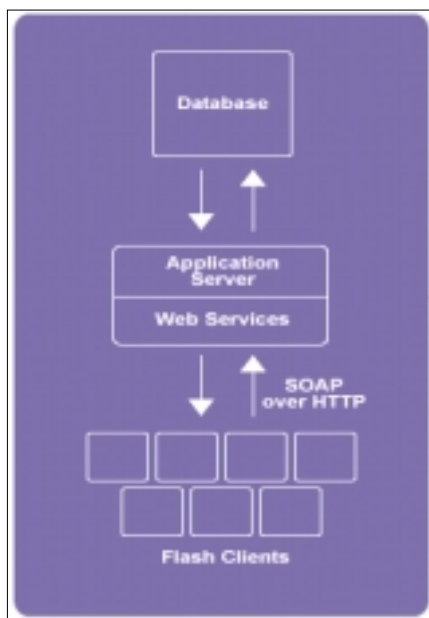


Figure 1: Flash accessing Web services

I'll be making reference to the `WebServiceClasses` component and not the `WebServiceConnector` component. The `WebServiceClasses` component is the minimum set of classes necessary to handle SOAP transactions programmatically, while the `WebServiceConnector` is more of a visual aid for accessing Web services from Flash. I believe in using a programmatic approach because it gives developers more control over their applications, allows them to better understand how the API works, and helps reduce file size.

I'm currently working for a company named *SensorLogic* and we're in the process of rewriting our Web portal to utilize Flash and Web services. *SensorLogic* is a machine-to-machine (M2M) ASP, and it's imperative that the user experience be as quick and reliable as possible. With the ASP model in mind, we'd like to expose certain services to our customers and resellers so they can build their own applications around our back-end system. We wanted our system to be flexible enough to serve data to Flash, ColdFusion, .NET, or Java. Web services gave us this flexibility. Figure 1 illustrates a basic communications architecture for Flash with Web services.

Although very extensible and flexible, Web services in Flash do have their limitations and drawbacks. The first and foremost is the overhead that SOAP transactions introduce. XML-based protocols are very descriptive, which means they add to

code bloat because of markup tags and headers. With this overhead, data usually takes longer to get to the Flash client and the XML has to be parsed, which slows the process down even more. This is definitely *not* recommended for large data sets or complex objects. On the other hand, if you're dealing with smaller data sets and simple objects, then Web services may be the perfect fit.

Another issue worth considering is security. Unless you plan on implementing SSL, all data sent over the wire is unencrypted. So if your application contains sensitive data, then enforcing secure transactions is a must. In Flash MX 2004 a new security measure was introduced, known as the `crossdomain.xml` policy file. This file basically says that a Flash client coming from domain A has authority to access services on domain B. This file resides in the Web root of domain B and is automatically read by the Flash client when any Web services are called.

To further explain this issue, let's say we want to write a Flash component that accesses a weather Web service and displays the temperature for a specified locale. This will be a problem if we're not authorized to access the remote server through the `crossdomain` file. In most cases it won't even exist on the remote server. In a controlled environment this isn't a problem, however. If we have control of the server that provides the weather Web service, we can add the `crossdomain` file to grant access to authorized clients. For more information on configuring `crossdomain` files, visit www.macromedia.com/devnet/mx/flash/articles/fplayer_security_03.html.

Flash Remoting

With all the hype around the new Web service classes, developers seem to think Macromedia has forgotten about Remoting. This couldn't be further from the truth. The Flash Remoting components for ActionScript 2.0 were recently released – and they're larger than life. Nothing has changed in the underlying

architecture of Remoting; the new components are just friendlier to ActionScript 2.0 developers. While Macromedia put a good amount of resources into the new Web service classes, this doesn't eliminate the need for Remoting.

If you've read any of my past *ColdFusion Developer's Journal* articles, then you probably know how much I rant and rave about Remoting. Yes, I'm a bit biased toward Remoting, but that's because I love CF and how nicely it integrates with Flash. All data that is transmitted between the Flash client and the application is serialized using the Action Message Format (AMF). AMF is a streamlined binary protocol that was modeled after SOAP. Flash Remoting automatically takes care of data type conversion from the server to the client and back again. Figure 2 illustrates a basic communications architecture with Flash Remoting.

Notice how Web services can still be introduced into this architecture. The application server can serve as a proxy between Flash and the Web service. An application server such as ColdFusion would invoke the remote method and return the data to the Flash client using Remoting. Taking this approach circumvents the `crossdomain.xml` security restriction.

As mentioned earlier, if you're dealing with large data sets and complex objects, then Flash Remoting is definitely worth a look. In most cases your performance will be better. If you're doing simple transactions such as retrieving stock quotes and weather data, then performance will be about the same or the difference will be too negligible to even notice.

One of the major drawbacks of Flash Remoting is the cost. The license runs \$999 for a single-CPU license at the time of this writing. This is not an issue for CFMX developers (or Java/JSP developers using JRun 4) because Flash Remoting comes built in! If you are building enterprise applications with Flash but you're not on a CFMX server, then purchasing a Remoting license is a no-brainer. I can say

**"Although very extensible and flexible,
Web services in Flash do have their
limitations and drawbacks"**

from experience that it's worth every single penny for the performance you'll be gaining. Okay, enough talk; let's take a quick look at the application.

The File-Browsing Application

Although I've spent a good amount of your time talking about each technology, I'd like to walk you through a basic file-browsing application. If you'd like to download the source code for this application, please feel free to do so at www.sys-con.com/coldfusion/sourcecfm. View the README file to

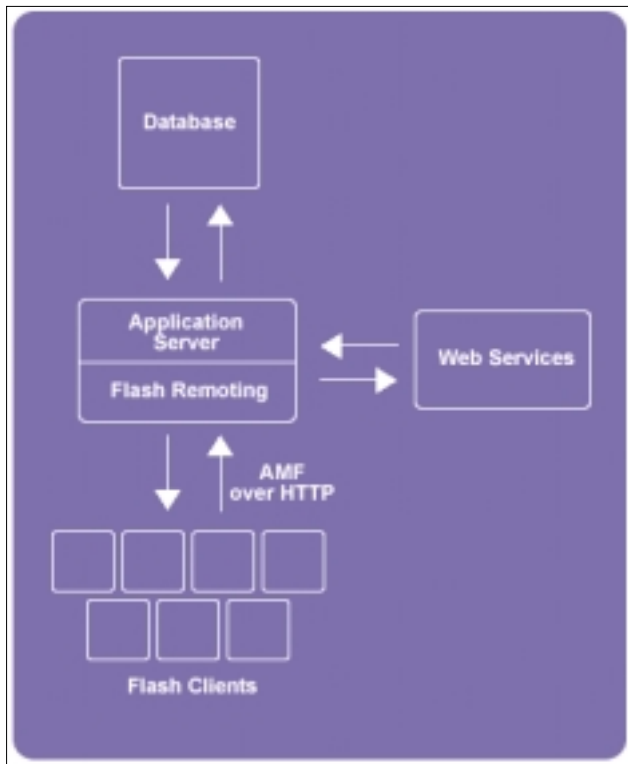


Figure 2: A basic Flash Remoting communications architecture

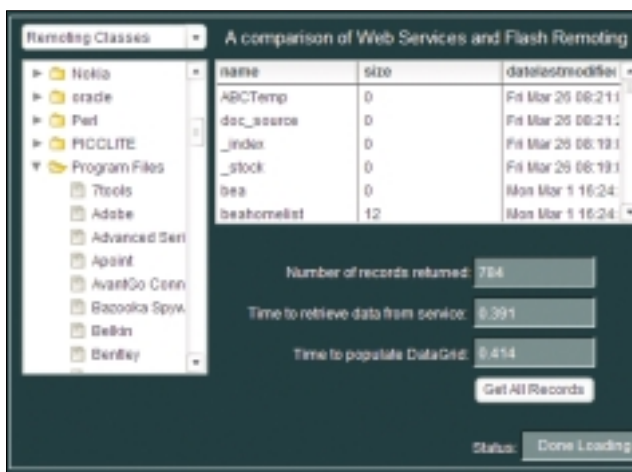


Figure 3: Screenshot of Web services and Flash Remoting application

see how to configure the application. The application consists of a Tree component that allows you to select a folder and display its contents in a DataGrid component. Each time a tree node is clicked, a service call is made to the server for data. The ComboBox controls whether the service call is made using Web services or Flash Remoting. Figure 3 shows a screenshot of the application at work.

Web service and Flash Remoting objects are created with ActionScript in a similar manner. After the objects are created, a service method called `getFiles` is invoked. This method resides in a CFC called `fileBrowser.cfc`, which can be invoked through a Web service call and also through Flash Remoting. I want to take a quick second to say how truly amazing CF is. In less than 15 lines of code I can expose both a Web service and Flash Remoting method that can be consumed by a Flash client. If you've had experience exposing remote methods with any other technology, then you can truly appreciate how simple CF makes this.

After playing around with the application there are a couple of things that you'll most likely notice. Overall, the performance of Flash Remoting will be better than that of Web services. When retrieving a small number of data sets the difference will be negligible and not even worth noting. That's why the "Get All Records" button was added, because I think this truly demonstrates what I discussed earlier. Since we're dealing with what could be considered fairly complex objects (recordsets) and large amounts of data, you'll see that Flash Remoting truly outperforms Web services. While testing on my local machine it generally took the Web service classes 4–5 seconds to retrieve and display 783 records, while it took Flash Remoting less than a second. Play around with it and your results may vary!

Conclusion

In this article I touched the surface of Web services and Flash Remoting with a few pros and cons of each. There's definitely enough information on these two topics that an entire book could be devoted to them. I encourage you to download the sample application and test it, tweak it, and break it. I hope it will ultimately help give you enough information to decide which technology fits your needs, maybe one or the other, maybe both.

One area that I did not have a chance to touch upon is debugging. If you're interested in learning more, I recommend you take a look at the Log class for Web services and the NetConnection Debugger for Flash Remoting. If you'd like to learn more about the NetConnection Debugger, please check out my earlier article in *CFDJ*, Volume 4, issue 10. All in all, the future is bright for Flash and ColdFusion developers. Stay tuned!

About the Author

Dennis Baldwin is a software developer for SensorLogic Inc, an M2M application service provider. He also runs and maintains an online community for Flash and ColdFusion developers at www.devvmx.com.

dbaldwin@sensorlogic.com



Dedicated Server Packages Starting at \$189/mo

All dedicated servers include:

- ▶ FREE STATS SOFTWARE!
- ▶ No long term commitments!
- ▶ FREE SQL server access!
- ▶ FREE MAIL SOFTWARE!
- ▶ Fair and simple pricing!
- ▶ Optional server maintenance!

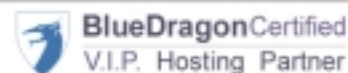
As one of the premier ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to provide **better service** to ensure your satisfaction. Not only do we offer the **finest in shared hosting plans**, but we now offer the **finest in 100% dedicated server plans**! Now you can afford the freedom of having your own dedicated server!

When your needs have outgrown shared hosting look to CFDynamics for **total freedom**. With dedicated server packages they're not offering an oxymoron; "virtually private" or "virtually dedicated" is **NEITHER** private nor dedicated. CFDynamics offers a solution that is **100% completely dedicated**. They don't play games with the fake stuff; CFDynamics only offers the real deal. Real Service. Real Satisfaction. Real Value.

Real Freedom.



PaperThin Partner



Visit us online or call to order!



Using a Data Classification Digital Taxonomy

As the volume of digital content grows, so does the need for efficient and accurate data retrieval

As an organization's vast collection of data continues to grow, it becomes increasingly difficult for users to find the information they need. You need only to look at the success of Google to see the importance of search engine technology. Unfortunately, traditional search engines that rely primarily on keyword matching often return unintended results.

This makes finding the information that you're really after time consuming and inefficient. To make data search and discovery more productive, organizations are turning to taxonomy-based data classification.

Taxonomy classification is a means of creating order out of large collections of data. At its most basic level, taxonomy is simply a collection of terms or subjects. The strength of the model, however, comes from the taxonomy's ability to also define a term's relationship to other terms. This provides the means to derive the terms' context based on the relationships in the taxonomy. If a single term has several different meanings, it will have these additional associations defined in the taxonomy.

In most implementations the model is flexible, allowing for relationships to be expressed in much greater detail than is available from a strict hierarchical model. This allows for the definition of "related" and "equivalent" terms; something that is more difficult in typical hierarchical trees. Among other benefits, this makes it possible to implement an Amazon-like "recommenda-



By David Athey

tion" engine to find related items that are defined in similar topic areas within the same taxonomy.

When a taxonomy classification is being used, data that is added to a system is classified using the terms that have already been defined in the taxonomy. When data is associated with one or more terms, the data inherits the properties and relationships of those terms. This reduces the work involved with classifying new data. Also, as the taxonomy definition is improved and updated, the new data associations will be effective for the existing data without the need to go back and manually reclassify it.

Finding Your Data: No Problem!

Removing keyword ambiguity should be a goal of all search implementations. With conventional search engines, a keyword search for the term "star" could return results both on astronomy as well as Hollywood actors. With a taxonomy-based search, the multiple contexts would be known and presented to the user allowing for further refinement of search results based only on the desired subject matter, or "facets." Irrelevant data is filtered out, leaving behind only the results that are applicable to the selected topic area.

The same system that removes ambiguity also allows for the benefit of data discovery. Looking for data using a taxonomy navigation tool is similar to browsing the book aisles of a library. You may not know what you're looking for, but you'll know it when you see it. Users are able to "browse" the data that is associated with nearby terms in the taxonomy, allowing them to find information they might not have discovered in a search using known keywords.

Given that the same term could be relevant to many different subject areas, there are potentially many paths to the same



Experience

Connect

Learn

Develop

Exchange

Macromedia MAX 2004

The Power of Experience

Waves are the powerful result of critical mass, various elements coming together at a specific place and time. The experience of people coming together to shape ideas and share information is equally powerful. That's MAX: a powerful experience.

Learn the best techniques from industry experts; get current on new products; and share ideas in a forum dedicated to delivering the next wave of great websites and applications.

Experience

Choose from over 80 different hands-on and workshop sessions, in seven tracks, to create a schedule to meet your specific needs.

Connect

Share ideas with leading developers and designers at networking receptions, Birds-of-a-Feather sessions, and a special event. Retain your competitive edge at general sessions where you'll hear directly from Macromedia® and other industry leaders on what's next.

It all happens November 1-4 in New Orleans. Be a part of it. Register today.
www.macromedia.com/go/max

Silver Sponsors

DoubleClick

 **HostMySite.com**


macromedia
PRESS

 **VitalStream**

MAX

The 2004 Macromedia® Conference

© 2004 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.

data, allowing for expansive data discovery. Imagine searching for a brand of merlot red wine and then being presented a selection of foods that go best with that variety. That is the power of a taxonomy classification based search!

Enter ColdFusion and XML

XML is the emerging standard for defining taxonomies. Many of the currently available tools for creating a taxonomy specification provide XML export functionality. This is good news for ColdFusion developers, who already have a collection of functions available for working with XML.

In an XML definition, each term in the taxonomy is an element with its own collection of attributes and subelements. A standard definition will include tag markup for each type of relationship that can be represented. For most terms this will include "narrower term" and "broader term" tags, indicating the term's hierarchical position in a given context. In more advanced systems, XML elements would also be added to represent the nonhierarchical relationships. A sample XML specification is shown in Listing 1.

Although XML is widely used as the language for taxonomy definition, an authoritative-format standard for this definition is still pending. Given this, it is best to make the implementation as flexible as possible, allowing for future attributes and term relationship types to be added easily with little to no refactoring. Ideally, an accepted Document Type Definition (DTD) will be created that allows for the validation of the XML. Until then, it is possible to implement custom validation that uses XMLSearch() with an XPath expression to validate the required XML elements. Organizations may also want to con-

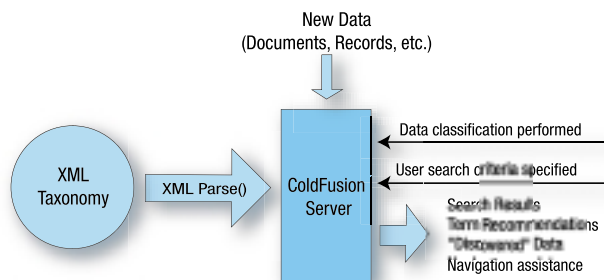


Figure 1: Process flow for a taxonomy search

sider creating their own DTD to be used for the validation.

Given a plain XML definition, it becomes trivial to use ColdFusion's XMLParse() to load the definition and create the XML object in memory. Once the XML object is obtained, an XPath expression can be used with ColdFusion's XMLSearch() to extract the relationships. Depending on the criteria specified in the XPath expression, it is possible to process a single specified term or the entire taxonomy at once. (See *CFDJ*, Vol. 4, issue 4 for an excellent article on parsing XML.)

Can ColdFusion Handle It?

Organizations that are most likely to implement a taxonomy classification system are those with high volumes of digital data.

Given the large amount of data, it becomes important to keep performance in mind when designing the system. The two main factors that effect scalability are the number of terms in the taxonomy and the amount of data associated with those terms.

A taxonomy with 20,000 or more terms is generally considered large. Parsing the XML and storing the terms into memory are potentially intense processes. With ColdFusion, however, tests using a 100,000 term taxonomy on a mid-powered server resulted in load times of only a few seconds. This included the XMLParse() call to create the XML object, the XMLSearch() call to retrieve the terms, and multiple assignment calls to create an associative array of the terms along with the defined relationships. An additional step to perform the validation added only a marginal increase to the total processing time.

Even though a taxonomy will typically be limited to several thousand terms, there may be millions of data records associated with those terms. Once finalized, the size of the taxonomy definition tends to stay more or less fixed, unlike the data count of an active system, which will see continued growth. Even though reading the data associations from memory is fast, the memory consumption could become excessive as the system ages. It is usually safe to load the entire taxonomy into memory, because even a large classification will make only a modest dent in memory consumption. This is not true with the actual system data in which the typical design tradeoff between speed and memory must be considered.

To avoid scalability problems, you can rely on a simple CFQUERY database call to retrieve the associations given to a specific term. For further improvements, commonly referenced terms and their respective data associations can be cached for fast lookup. See Figure 1 for a basic process flow starting with the initial XML import, and concluding with the user obtaining results based on the specified criteria.

Future Development


The use of a taxonomy classification system for digital data is still relatively new. Over the past five years there has been much progress. However, work is still needed before there is a widely accepted vocabulary and common understanding of the framework and concepts.

One of the biggest challenges for an organization that wants to implement a taxonomy classification is the time and effort involved in creating the definition specification. Currently there are some commercially available definitions, but these are offered only in a limited number of business areas. Organizations that already have an institutional thesaurus are well positioned to use taxonomy-based classification. A thesaurus is often a precursor to a taxonomy, and the terms and vocabulary used to create a thesaurus are easily transferable. The National Information Standards Organization (NISO) has published guidelines for the construction of a Monolingual Thesauri, which is available in the ANSI Z39 specification. This is a good starting point for those exploring the possibility of implementing such a system.

Another implementation challenge is ensuring that data is classified correctly. There are auto-classification tools available that attempt to derive data context by using natural-language algorithms. These tools attempt to "understand" the content of

the given data by evaluating not just the keywords, but also the circumstance. Once attained, the tools will assign the data to the proper term in the taxonomy. The accuracy of these tools won't match human classification but could be acceptable especially if the data is already tagged using some form of metadata.

Gaining Steam...

The idea of using a taxonomy to organize and classify data is not new. In fact, the term "taxonomy" comes from biology in reference to the classification of living things. Applying this idea to vast stores of digital content, however, is a practice that has only recently gained steam. As digital content repositories grow, finding your target data quickly and accurately will seem more like finding the proverbial needle in a haystack. A taxonomy classification system is an excellent complement to a traditional keyword search and will help users efficiently find the data they need. 

About the Author

David Athey is a senior developer with PaperThin Inc. based in Quincy, MA. He is an Advanced Certified ColdFusion Developer with expertise in Enterprise Content Management and Web-based publishing.

dathey@paperthin.com

Listing 1: A sample taxonomy XML specification for classifying wines by type and region

```
<taxonomy version="1.0">
  <name>An XML Taxonomy Specification</name>

  <!-- Define a topic area -->
  <facet>
    <name>Wine Varieties</name>
  </facet>

  <term>
    <name>Red Wine</name>

    <!-- Additional term properties -->
    <annotation type="sn">Red wine is good for cooking</annotation>

    <!-- Narrower term designations -->
    <nt>Pinot Noir</nt>
    <nt>Merlot</nt>
    <nt>Cabernet Sauvignon</nt>
  </term>

  <!-- Define a topic area -->
  <facet>
    <name>Geographic Regions</name>
  </facet>

  <term>
    <name>Italy</name>

    <!-- Additional term properties -->
    <npt type="synonym">Italian</npt>

    <!-- Narrower term designations -->
    <nt>Tuscany</nt>
    <nt>Piedmont</nt>
    <nt>Adige</nt>
  </term>
</taxonomy>
```

Download the Code...
Go to www.coldfusionjournal.com



It's everybody's PDF™

How would you like to provide PDF generation and manipulation capabilities to your entire organization without breaking the bank? activePDF offers affordable server-based solutions that take the guesswork out of the PDF conversion process making it both simple and seamless to end users.



► activePDF Server™ is a server-side PDF print driver, capable of redirecting output from virtually any Windows application to PDF. With over 75 properties and methods available from its unique COM interface, activePDF Server provides you full control over your PDF output on a job-by-job basis.



► activePDF DocConverter™ enables you to convert over 280 different file formats directly to PDF. DocConverter can be used in batch mode, via "watched folders", or in client/server applications via a distributable remote submission component.



► activePDF WebGrabber™ dynamically converts any URL, HTML stream or HTML file to PDF on the fly, while retaining embedded styles. Custom page break tags, built-in page numbering, and dynamic headers and footers provide additional control over PDF output.



► activePDF Toolkit™ is a programmable COM object that simplifies PDF manipulation. Licensed per server, Toolkit enables you to append, stamp, stitch, merge, paint, secure, form-fill PDFs and more.



Copyright © 2004, activePDF, Inc. All Rights Reserved. "ACTIVEPDF", "Leading the iPaper Revolution", "It's Everybody's PDF" and the activePDF logo are registered trademarks of activePDF, Inc. All activePDF product names are trademarks of activePDF, Inc.

Three Steps to Better Leadership

Start with yourself

Successful leadership is the most important best practice a development team can adopt.

Have you ever been on a team that missed its target deadlines, launched critical programs only to later find they had defects, went over budget, or just plain failed? If you have never experienced any of these typical software development problems, consider yourself extremely lucky and think about (and thank) your team leader.

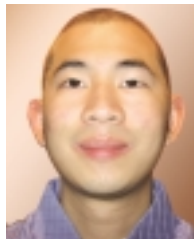
There are many established best practices to mitigate risks such as slipping timelines and budget-breaking costs. Requirements gathering, architectural design, code inspections, and quality assurance are all techniques proven to help, but what helps one team effectively adopt these concepts while another fails to implement any of them successfully? In *Rapid Development*, Steve McConnell asserts that “collectively, peopleware issues matter more than process, product, or technology.” Successful leaders align these four tenets of software development: people, process, product, and technology.

In some cases the leader is the group or project manager; in others he or she may be a senior programmer acting in a technical lead role. Regardless of the team structure there will typically be one person (sometimes several) responsible for the team's success. If you are in that position, or would like to be in that position, there are three steps you can take that will increase your leadership ability and maximize your team's chance for success.

The first step to becoming a better leader is learning how to lead yourself. The second step is learning how to lead individuals. The third step is learning how to lead a group of people as an effective team. Each step should be completed successfully before taking the next. Like a pyramid, the capstone stands higher when put on a foundational base (see Figure 1). When you build on the steps you have completed, each step will be stronger and take you farther. This is the foundation upon which strong leadership skills are built.

Step 1 – Leading Yourself

As I mentioned earlier, the first step to becoming a better



By Reuben Poon

leader is learning how to lead yourself. Leadership expert John C. Maxwell quips, “If you wouldn't follow yourself, why would anyone want to follow you?” Developing yourself as a leader also makes you a better follower, and all leaders still follow someone else. There are four elements of leading yourself that create the foundation for successful leadership: character, competency, chemistry, and celebration.

The first foundational element – character – is essential to building trust. Only when people trust you will they follow you. You gain trust by being a person of character and acting with integrity. Integrity means firmly adhering to a code of values; when you have integrity, you do what you say and say what you do. If your boss were to constantly lie to you, would you be more or less willing to follow him or her?

After trusting their leader, people need confidence in their leader's ability – competence. In the movie *Gladiator*, people followed Maximus because they had seen his accomplishment and therefore had confidence in his ability. Everyone has unique strengths that give them the potential to not only be competent, but to be excellent. Excellence will attract people to follow you.

Excellence comes from identifying your strengths. Strengths are what you do best. As a programmer, your strength might be that you can create order out of thousands of requirements. You might have a knack for meticulous attention to detail. You might have an inquisitive mind that helps you ask the right questions to foresee problems. In *What Makes an Effective Executive*, Peter Drucker explains that, “effective executives try to focus on [what] they'll do especially well.” People using their strengths average 67% better productivity than those who concentrate on fixing their weaknesses. Can you imagine the possibilities if you could finish your normal day's work in a little over half a day?

Your job is to find out what your strengths are and build on them. The Gallup Organization's StrengthsFinder (www.Strengthsfinder.com) is a tool that can help you figure out where your potential for excellence lies. It is only by working in your area of strength that you can become an excellent leader.

Because leadership requires working with other people, you cannot rely on character and competency alone but in

connection with chemistry – how you relate to and work with other people. By understanding how you relate to people and the world you will be better prepared to work with others and help them work with you, building team chemistry. Tools such as the Myers Briggs Type Indicator can help you take a look at your personality, how you communicate with others, and what you can do to leverage that knowledge (see “Assessment Tools” sidebar). How much faster would projects be completed if communication challenges were minimized?

Congratulations! You have taken your first step to becoming a better leader. By building your foundation on good character, excellent competence, and chemistry with others you are well on your way to leading others. Before we take the next step, let me introduce you to the fourth “C” – celebration. Great leaders publicly celebrate their teams’ achievements and accomplishments as well as their own victories. So call up your best friend and tell them you took the first step toward becoming a leader!

Step 2 – Leading Individuals

Now that you have successfully led yourself, you are ready for the second step – learning to lead another. To successfully lead another person, you have to give that person a reason to follow you, help him or her be successful while following you, and give him or her a reason to keep following you.

If you are the technical lead or manager, programmers on your team are supposed to follow your suggestions, but things might not always turn out that way. Perhaps you are not in a leadership position, but most of the people on your team follow your suggestions. Why is that? In both scenarios the answer is one word: Influence. John C. Maxwell defines leadership as influence; nothing more, nothing less. True leadership is when people follow you because they want to.

When people want to follow you, you have the ability to accomplish so much more. However, before you can jump into getting people to follow you, you first have to look at each individual. Do you care about this person or do you just see him or her as a means to your ends? Unless you care about a person and his or her personal successes, you will never be able to effectively lead that person because you will always be someone he or she has to follow. Once you care about an individual’s success as much as (or more than) your own success, you need to show it. Treat that person as if he or she does not have to follow you and focus your attention on him or her. Peter Drucker suggests, “Think and say ‘we.’” Make the other person’s needs more important than your own. Put your trust and belief in that person. People have a funny way of living up to your expectations. Finally, add value to his or her life. If a person’s life is better because of you, then that person will not only follow you but will work his or her heart out for you.

How do you show a person that you value him or her? Make that person successful – help develop the individual’s potential. Remember the first step you took? Take him or her through that same process. Help the individual be a person of character and integrity. Show that person his or her strengths and how to become a better programmer by using those strengths every day. Help the person realize his or her unique characteristics and how the two of you can better communi-

cate and work together. Most important, celebrate your shared victories.

Now that you are both successful, you want to keep it that way. There are a few “people management” basics that can help you lead more effectively. Two of the most important things you can do are to set clear expectations for your desired outcome and to give the developers on your team what they need to do their job.

What comes next? Get out there and do it! The best way to learn leadership is to practice it. Remember, just because you are not in a position of leadership does not mean that you cannot lead others. Care about other people’s success. Add value to their lives. Not only will that make you a better leader, you will be a better person.

Step 3 – Leading Teams

What happens when you influence several people and they are all following you? You’ve got a team! Your effectiveness as an employee multiplies because you can now leverage the strengths of an entire team of people. There are four things to do when leading a team: build the team, keep the team healthy, direct the team, and raise more leaders.

In Steps 1 and 2 you built a foundation of competency by maximizing your strengths. Teamwork (Step 3) is where you will see exponential rewards for that work. In the context of a team you can share your individual strengths to make your output greater than the sum of the individual parts.

Let’s examine a hypothetical programming team: Larry, Moe, and Curly. Larry is the senior programmer, has built influence with Moe and Curly, and is now the technical lead. Larry knows his strengths (Step 1) and the strengths of Moe and Curly (Step 2). Moe is very detail oriented and approaches

Practice Leading

The only way to learn leadership is by practicing it. Let yourself progress naturally through the three stages of learning leadership. One of the best ways to do this is by volunteering somewhere. In a volunteer organization, leadership in its purest form is required. Leaders don’t have the traditional mechanisms to leverage, such as money or the ability to fire someone. Leaders have to lead by influence alone. If you can become a successful leader as a volunteer, you will probably become a very successful leader at work.

Assessment Tools

The aforementioned tools are just that – tools. However, I have found them to be very good at providing the basics of what is needed to help take potential leaders through this first step. The most important aspect of choosing a tool is consistency. Once you pick it, use it! When the majority of people in your organization speak the same language it is much easier for them to learn how to work together and for leaders to hit the ground running when a new member joins their team. For more information on these assessment tools, visit www.TheNehemiahTeam.com/resources.

Once you're in it...



...reprint it!

- ColdFusion Developer's Journal
- Java Developer's Journal
- Web Services Journal
- Wireless Business & Technology
- MX Developer's Journal
- PowerBuilder Developer's Journal
- .NET Developer's Journal
- LinuxWorld Magazine
- WebSphere Journal
- WLDJ

Contact Kristin Kuhnle
201 802-3026
kristin@sys-con.com

REprints



programming

all things with a methodical, disciplined manner. Curly's strength is in strategically figuring out the best approach to solving a problem or writing code.

Based on this knowledge, Larry can share the team's strengths. He might ask Curly to lead the team to figure out which framework to use or how to best balance the design of the system. Because Larry knows that he and Curly are not detail oriented, Larry can ask Moe to create and maintain requirements documents because Moe loves attention to detail. This basic example shows you how you can leverage the strengths of your own teammates and manage one another's weaknesses. Imagine what your team could do if your strengths are pooled and your individual weaknesses are minimized.

Now that the team is running at maximum capacity, you have to make sure the team stays healthy. We are not talking about medical health (although that is important) but team health. Crucial elements of a healthy team are trust between team members, the ability to have healthy debates that lead to better decisions, people being held accountable, and focus on collective achievements. The team leader is responsible for facilitating these conditions.

Successful leaders are often tasked with making hard decisions. These decisions decide the direction of your team. As the leader, making these decisions is your primary responsibility. As you do this, it is important that you are clear about your team's direction. More important than making sure you are headed in the right direction is making sure everyone is going

"Excellence comes from identifying your strengths"



Figure 1: Leadership steps

in the same direction. It will be much easier to change course if everyone is together! Communication is crucial to making this happen. Make it a goal to over-communicate decisions and direction to your team.

Finally, your most important task as a leader will be to raise up new leaders. In *Good to Great*, Jim Collins explains how all of the good-to-great companies outlived the company leader: "[Great] leaders set up their successors for even greater success in the next generation." When you create more leaders on your team, you exponentially increase your opportunities and what your team can do.

Whether you are a programmer, a technical lead, or a group manager, you will increase the potential for success by learning how to lead yourself, how to lead individuals, and how to lead teams. Remember that your leadership ability determines not only your effectiveness on the job but also your team's effectiveness and, ultimately, the success of both. Effective leadership will help your team finish projects on time, with fewer defects, and under budget. Why not take your first step to becoming a better leader today?

About the Author

Reuben Poon has previously written for CFDJ and is now working with IT teams to develop their employees' leadership potential. He is president of The Nehemiah Team and helps companies develop their leaders and leverage their employees' strengths. Reuben would love to receive your feedback on this article. For more information on developing your company's leaders, go to www.TheNehemiahTeam.com.

rpoon@TheNehemiahTeam.com

Complete source code and asset management in Dreamweaver MX—now possible with Surround SCM.

Dreamweaver users know a beautiful Web-based product is only skin deep. Underneath, it's a tangle of hundreds or thousands of ever changing source files. Without a good development process and strong tools, bad things happen. Surround SCM can help.

Surround SCM lets you...

Track multiple versions of your source files and easily compare and merge source code changes.

Check out files for exclusive use or work in private workspaces when collaborating on a team.

Automatically notify team members of changes to source files—push changes through your organization.

View complete audit trails of which files changed, why, and by whom.

Associate source code changes with feature requests, defects or change requests (requires additional purchase of TestTrack Pro).

Remotely access your source code repository from Dreamweaver MX.

Surround SCM adds flexible source code and digital asset control, powerful version control, and secure remote file access to Dreamweaver MX. Whether you are a team of one or one hundred, Surround SCM makes it easier to manage your source files, letting you focus on creating beautiful Web-based products.

Features:

Complete source code and digital asset control with private workspaces, automatic merging, role-based security and more.

IDE integration with Dreamweaver MX, JBuilder, Visual Studio, and other leading Web development tools.

Fast and secure remote access to your source files—work from anywhere.

Advanced branching and email notifications put you in complete control of your process.

External application triggers let you integrate Surround SCM into your Web site and product development processes.

Support for comprehensive issue management with TestTrack Pro—link changes to change requests, bug reports, feature requests and more.

Scalable and reliable cross-platform, client/server solution supports Windows, Linux, Solaris, and Mac OS X.

Want to learn more?



Download Seapine's just-released white paper, **Change Management and Dreamweaver**, to discover tips, tricks and best practices for achieving full software configuration functionality. Visit www.seapine.com/whitepaper.php and enter code WM0604 to receive your copy today.

www.seapine.com
1-888-683-6456




Looking over the session descriptions and presenters currently listed on the Macromedia Web site (and ignoring the hands-on sessions since they're almost all already filled), here are just a few of the sessions that I would recommend (in no particular order):

- **Mobile SMS Applications Made Easy (Damon Cooper):** Damon is the director of engineering for ColdFusion at Macromedia, a very sharp guy, and this is one of the new features in Blackstone... need I say more?
- **Creating Flex Components (Brandon Purcell):** Whether or not you're into Flex, Brandon really knows his stuff (J2EE, ColdFusion, and Flex) and is sure to be technical.
- **The Future of ColdFusion: Blackstone (Tim Buntel):** Tim is the product manager for ColdFusion at Macromedia and is extremely charismatic when he presents.
- **The Infrastructure Impact (Dave Watts):** Dave is CTO at Fig Leaf Software and is one of those guys you should go see no matter what he's talking about. Dave is also the only presenter I know who can put the audience on the spot rather than the other way around.
- **Building Internationalized and Multi-Lingual Applications (Steve Drucker):** Steve is CEO of Fig Leaf Software and is another one of those guys you should see no matter what the topic. In addition to whatever he's talking about you're usually sure to learn how to do something very wacky with JavaScript. He is the JavaScript Ninja.
- **Object Oriented ColdFusion (Hal Helms):** Hal is the mas-

termind behind Fusebox and MACH II. Whether you like those methodologies or not, he's one of the sharpest guys around. Hal always offers interesting insight and is yet another of those guys whose presentations are not to be missed.

- **ColdFusion Printing and Reporting (Dean Harmon and Xu Chen):** Dean and Xu are on the ColdFusion development team (I believe Dean played a part in the reporting functionality and Xu played a part in the document generation functionality). I've never seen them present but how can anyone miss a chance to listen to a couple of ColdFusion development team members talk about new features in Blackstone?
- **Building a Basic CMS with ColdFusion and Dreamweaver (Dave Gallerizo):** VP in charge of consulting, Dave, like the other folks I've mentioned from Fig Leaf, is a highly rated Macromedia Certified Instructor. No matter what the topic, Dave always knows his stuff and has a terrific presentation style.
- **The Macromedia Vision for Lighting Up a Billion Mobile Devices (Anup Murarka):** This session takes the award for "coolest title". I've never heard Anup speak but he's senior director and responsible for marketing mobile and devices at Macromedia, and this is a subject area that is becoming very hot.
- **QStructured Development, ColdFusion Done the Right Way (Ben Forta):** Ben is Macromedia's Technology Evangelist and also happens to be a terrific speaker. Besides, who can resist that English accent?
- **QBuilding Your First Dynamic Web Application (Sue Hove):** Sue is in charge of instructor readiness at Macromedia Training. In fact, I first certified as a Macromedia instructor in front of Sue. I've seen Sue speak several times – she's an unbelievable presenter and instructor.

Also noteworthy are several very interesting-sounding Flash Cast topics, presentations by several other ColdFusion gurus such as Ray Camden ("Coding for Reuse"), Sam Neff ("Integrating with Microsoft Office"), and Jeff Tapper ("Using ColdFusion to Power Flex and Flash Applications"), and so much more. I don't have enough space to name them all, so I drew on years of experience attending these conferences and tried to highlight some of the sessions I think people would most regret missing. The sessions I've listed are all 60-minute lectures except for Sue Hove's and Ben Forta's. I generally recommend attending any hands-on session if you have a chance. I should also mention that as usual I'll be presenting at the conference ("Advanced ColdFusion Components and Web Services"), so if you're there and you see me, be sure to stop by and say "hi."

Like I said, in addition to the speakers and topics I've named here, there are so many other terrific speakers and topics to be seen at MAX. I didn't list all of them, but if you're at the conference try to take in as much as you can; you won't regret it. If you were unable to attend MAX, I have good news for you: be sure to read next month's issue of **CFDJ** because it promises to have the most comprehensive post-conference summary in **CFDJ** history! 

Don't Miss CFDJ's Next Issue!



Comprehensive Post-show Coverage of MAX: Whether you attend or not, this will be an issue not to miss.

A Look at Ant: A new Ant deployment build file for a CMS.

CF101: ColdFusion Components: While functions are great for modularizing functionality, components allow you to modularize both data and functionality.



\$99
per person

CF_Underground VI

October 31st 2004

10:00am - 5:00pm

New Orleans, LA

Check website for exact location and details



www.cfconf.org/cf_underground6/

Fill your brain with lots of cool CF programming meta
tips & tricks, then relax and have a drink on us!

CF_Underground is the coolest pre-MAX
event you'll attend during your visit.

Pick the brains of several experts like

Simon Horwith, Sandra Clark, Shlomy

Gantz, Hal Helms, Michael Smith,

and more!

TeraTech Event

www.teratech.com

301.424.3903





CONTROLLING PAGE SEQUENCING WITH AN EVENT-DRIVEN STATE MACHINE

Write more robust, secure, and reusable code

One of the persistent headaches of Web application development is the developer's inability to control the sequence in which application "pages" are executed. The back button, refresh, bookmarks, double-clicks, and browsers' inconsistent page-caching behavior all create special problems that result in everything from "variable undefined" errors to duplicate data records.

To prevent these errors, developers have often resorted to passing around flag variables and status variables that make code hard to maintain over time. In this article, I explore a way to gain positive control of application page flow in server-side code, resulting in code that is more robust, more secure, and more reusable.

Controlling Page Sequencing with an Event-Driven State Machine

It has been said that the difference between an in-house application and commercial software is that for an in-house application, there has to be at least one right way to use the system. For commercial software, there can be no wrong ways to



By David Chandler

use it. The same can be said of intranet vs. Internet applications. This presents tremendous challenges in the Web environment, where the developer has little control over the sequence of user actions. Think of your application as consisting of all the code contained in individual pages (or fuseactions, if you're a Fusebox developer). Thanks to the back button, double-clicks, and bookmarks (not to mention spiders and URL scanning tools), users can run (request) all those little pieces of code in any order they choose. Is your code ready for this?

Background

In the early days of the Web, much ado was made about the performance and scalability that result from the stateless nature of HTTP. Unlike complicated client-server programming, you don't have to worry about maintaining connections to a server and managing the flow of control through an application. Indeed, this model works very well for Web sites, where pages may contain some dynamic content but are designed to be accessed in any sequence. However, dynamic Web applications often require that "pages" be visited in a particular order, such as an order entry wizard, shopping cart checkout, or online funds transfer procedure. In these applications, the intended page sequencing is often enforced only by the various HREFs and form actions. Few applications explicitly enforce the intended page sequencing. Thus, although there is one right way to use the application using the links and buttons on each page, lots of

unintended sequences are also possible. How can we control page sequencing when the user can enter an arbitrary URL in the address bar?

The fundamental problem is the URL. Its very name, Universal Resource Locator, suggests that a URL was designed to serve as an address to a particular object, such as a static Web page. Used this way, it does not matter in what order the user clicks on URLs. However, Web applications use URLs both to retrieve objects and to initiate actions via POST requests, such as placing an order or running a search. When used to initiate actions, sequencing is very important. Suppose, for example, that you are writing a banking application that allows a user to transfer funds between accounts. This may involve a wizard in which the user selects the accounts for the transfer, enters a transfer date and amount, and submits a final confirmation. You must make sure that the user progresses through the wizard pages in order and cannot go back in the sequence to alter the transaction in progress except via the paths you define. Unfortunately, neither HTTP nor HTML has a way to enforce the sequence in which users initiate actions through URLs. The back, refresh, and bookmark features make it all too easy for users to deviate from the intended sequencing.

There are various ways to handle these issues that arise with browser-based applications. You can open the application in a new window and disable browser features. You can write a Javascript event handler to disable the submit button after it has been clicked. You can look at HTTP_REFERER to ensure a form post comes from the expected page. But all these methods rely on browser features and are therefore not secure and/or are browser version specific. Some users disable Javascript, and proxy tools such as Paros make it all too easy to modify any part of a Web request including the HTTP_REFERER. Thus, you can't rely on Javascript or HTTP_REFERER to enforce page sequencing.

Ideally, we'd like a server-side solution to these problems that can be implemented in a modular fashion. We should not have to put an include file on every page or set custom flag variables for each restricted page sequence. In addition we'd like the ability to change in one place the way that the entire application handles the back button, and so forth. To do this, we need a framework that provides what the Web protocols left out: the ability to control the flow of pages. I'll call it application flow control.

Moving From Page Centric to Event Driven

To build a server-side flow control engine, we first need to rethink the page-centric model for Web applications. In the page-centric model, every form typically has its own action page that runs code to save the form data from the previous page as well as display the next page.

A Web application has more in common with a traditional client-server application than with a static Web site. A Web application, similar to a client-server application, typically consists of forms populated with database data. Forms have action buttons and familiar controls such as check boxes and text input fields. More important, a Web application, similar to a client-server app, is event driven. Each click on a submit button or hyperlink is an event that triggers some action code and possibly a transition to another form on the server. The address in the URL points to the event handler code.

However, here we come back to the key difference: In a client-server app, such as a Visual Basic desktop application, the user can only interact with the application's desktop user interface in the prescribed ways. The user cannot go directly to a different form in the application except by triggering a valid event on the current form. This is because the application is always aware of its state. A VB form only exposes itself to the user if, by interacting with the UI, the user triggers the events required to display the form. In a Web application, however, the user can bypass the UI and change the program state by entering a new URL (via the back button, a bookmark, etc.). To prevent a Web user from doing this, we must move the flow control logic away from the browser and back to the server. We would do this using a Web state machine.

(Re-) Introducing the Finite State Machine

Back in the marbled halls of your alma mater's computer science department, you probably learned about the finite state machine. In a nutshell, a state machine models program flow as a collection of states and state transitions that occur in response to events. At the heart of a state machine is a controller that is notified of events when they occur, executes some action code corresponding to the event (an event handler), and determines the next state based on the state transition map. This can be represented with a simple formula:

current state + event = action code + next state

In a Web application, we can think of each HTML page as a state. Each button click or hyperlink is an event that causes (a) some action code to run (the event handler) and (b) the browser to transition to another state. On the server, the Web state machine remembers what state (or view) the application is in so that when a new request (event) comes in, the state machine can determine whether the event is valid for the current state. If this is the case, the state machine runs the event handler and transitions to the next state defined for the event.

Designing the Web State Machine

The Model-View-Controller (MVC) and Front Controller design patterns provide a useful foundation for our state machine. In the Front Controller pattern, used by popular application frameworks such as Fusebox and Mach-II, all requests come through a single "page," such as index.cfm. This dispatches each request based on a URL parameter. Our state machine implements this idea with a "flow controller" module that keeps track of the user's current state (view) in the Session scope. It then uses the combination of current state and the event name in the URL to run the appropriate action code and include the view code for the next state.

When using a state machine, URLs no longer contain any information that can be used to go to a specific page. Instead, each URL contains only the address of the controller and an event name:

```
<A HREF="flowctl.cfm?event=addUser">
```

If no event is specified in the URL, the flow controller will simply show the view for the current state. If an event is not valid in the current state, the flow controller will send an error message and abort the request.

Our simple Web state machine consists of two files: a flow controller and a flow map. Flow_map.cfm (see Listing 1) maps each event to the corresponding action code and next state. The flow map is the heart of the state machine and will look very familiar to Fusebox developers. Unlike Fusebox, however, there are two layers of CFSWITCH/CFCASE statements. The outer switch selects the current state. For each valid state, a nested inner switch selects the current event. Each case in the inner switch includes the event handler (model) code for the event and assigns the name of the next state to Variables.nextState, which is used by the flow controller to update the state machine and include the corresponding view code.

Flowctl.cfm (see Listing 2) is the front controller that runs in response to each request and maintains the current state information. After setting up each request, the flow controller simply includes the flow map using <CFINCLUDE>. If the flow map sets a next state, the controller advances the state machine to the next state and includes the corresponding view code. This simple controller assumes a one-to-one correspondence between the name of each state and the name of the corresponding .cfm file to display the view.

How the Web State Machine Works

Let's follow a typical flow using the flow map for the order entry wizard shown in Listing 1. The example code uses the sample Northwind database that comes with Microsoft Access and SQL Server:

1. The user enters the flow by going to the URL flowctl.cfm. The controller creates a structure "sFlowData" in the Session scope used to hold the state machine's internal variables. In addition, the controller creates Session.sUserData and a reference to it named "thisFlow", which is used by the model and view pages to hold data the user enters. The controller sets the start state to the default "init" and the current event to the default "startFlow". The controller includes the flow map, which runs the initialization code in model/initFlow.cfm. The flow map sets the next state to "choose_customer" so the flow controller advances the state machine and includes the view code from views/choose_customer.cfm (see Listing 3).
2. On the "Choose Customer" view, the user chooses a customer from the select box and clicks the next button. Note that the form action in Listing 3 points to flowctl.cfm, and the form uses a submit button named "event". The text of the button is "next", so the flow controller gets a request with the variable Form.event = next. The controller includes the flow map, which in turn runs the action code model/act_saveCustomerSelection.cfm and sets the next state to "choose_products".
3. On the product selection screen (see Listing 4), the user has three choices: changeCustomer, selectProduct, or finish. You can see this by either looking at the events defined in the submit buttons or looking at the flow map in the switch nested inside the "choose_products" case. In the flow map, note that no matter which button the user presses in the "choose_products" state, the form data will be saved first. This is because the flow map includes the model code to save the form before looking at the event. This is where the

event-driven model really shines. In a page- or action-centric application, you would have to include the code to save the form in three different places. However, the event-driven state map neatly associates the code to save form data with the view that contains the form.

Rules for Using the Web State Machine

To implement this Web state machine example using the flow controller, your application must follow a few simple rules:

1. All HREFs and form actions must point to flowctl.cfm?event=event_name. All form actions must point to flowctl.cfm and must specify the event name either in the form action URL or using a named submit button as in the example code. If an event is invalid for the current state, the controller writes an error message and aborts the request. If no event is present, the controller includes the view code for the current state.
2. Put your initialization code in the "startFlow" event handler for the "init" state in the flow map. This will run once when the user first enters the flow by pointing to flowctl.cfm.
3. To reinitialize a flow and start over, use the URL "flowctl.cfm?event=startFlow".
4. Keep your action code separate from your view code. I like to think of it in terms of the default color-coding used by ColdFusion Studio. View code should contain HTML only, except for <CFOUTPUT> tags used to build the view and perhaps "view helper" queries to populate list boxes. Model code included for each event should contain CF tags only (red code).
5. The example controller includes view code for each state based on the state name. You can change the naming convention on the last line of the flow controller.
6. Use the variable "thisFlow" to persist form fields and other data used throughout a sequence of related pages (one page flow). This is especially powerful in more complicated controllers in which you have multiple flows. When you exit one flow and enter another, data from the previous flow is automatically cleared.

For more help using the example state machine, you can download all files in the example application from www.turbo-manage.com.

How Is a Web State Machine Different?

The Web state machine has much in common with frameworks that make use of the MVC design pattern and a front controller. The key difference in the Web state machine is the use of a stateful flow controller to enforce the sequence in which actions (or events) may be executed. Compare the earlier formula for a state machine with this formula implemented by stateless front controllers:

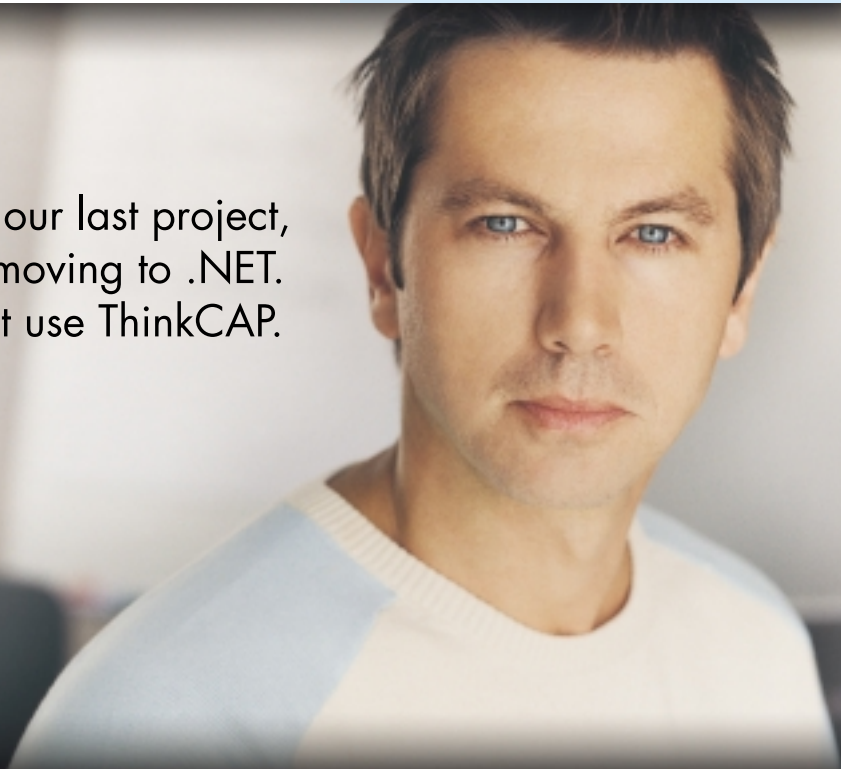
Event (or fuseaction) = action code + next state (view)

This formula omits consideration of current state on the left side. This is why a Fusebox map has only one layer with a case for each fuseaction, whereas the state machine flow map has nested switch statements. This way, events are available

Think .NET development is more productive than J2EE?

Think **again**.

Due to delivery pressures on our last project,
we thought about moving to .NET.
I suggested we stick with Java, but use ThinkCAP.



Think **better**.
ThinkCAP[™]

ClearNova's ThinkCAP is a comprehensive application platform that simplifies and accelerates the development and maintenance of J2EE-based business applications by 50 to 70%.

ThinkCAP's visual & intuitive designers bring high productivity to business developers (those with VB or PowerBuilder-like skills), content owners, and administrators while allowing J2EE architects & programmers to leverage its component infrastructure and build business logic using the tools and approaches they prefer. ThinkCAP utilizes existing infrastructure, web services, legacy systems, and business applications.

ThinkCAP saves organizations time and money—and lowers project risks. Applications are written faster and require less maintenance. Project teams utilize in-house skills and require less training. Existing infrastructure and application servers are leveraged.

With ThinkCAP you can build quality applications faster.

Learn more about ThinkCAP at www.clearnova.com/thinkcap



Highly Visual Development Environment

MVC Framework with Page Flow & Actions

Advanced Data Aware Controls:
Forms, DataViews, Queries, Navigations
Workflows, Graphs, Treeviews, Grids, Tabs

Smart Data Binding[™] to data, objects, XML
sessions, or requests

Browser & server-side validation

Visual unit testing with RapidTest[™]

Service Flow Designer aggregates Web
Services, EJBs, XML, and POJOs

Content Management engine & tools

Supports .NET clients

Integrated, seamless security

Use any app server or 3rd party tool

only within certain states. One further note: The term “event driven” does not necessarily imply the use of a state machine. Mach-II is an event-driven framework, but this term is used to describe the way it uses event listeners to decouple events from the code that executes in response to events.

Seeing the Invisible: Benefits of Using a Web State Machine

I first deployed a Web state machine to secure an application in which there were many possible entry and exit paths from every page. Over time, this had led to a proliferation of flag variables getting passed around between pages. These were used to keep track of where the user had been to save the correct form data from the preceding page. In all such applications there is a sort of map that indicates which pages lead to where, but the map is hidden in the URLs and form actions buried in all the pages.

First, the primary benefit of using a flow controller such as the one presented here is that it provides a single place to see how every page leads to every other page. Implicit dependencies that are hidden throughout your pages become explicit and visible in the flow map. You can easily search here to find all the events that lead to a given state, as well as all the events that lead from it. This makes it much easier to maintain applications with complex user interfaces. By refactoring your application into model code, view code, and a flow controller, you eliminate all the flag variables and conditional logic to track page sequencing. Your pages become more loosely coupled, and this promotes readability and reuse.

Second, a stateful flow controller improves application security by preventing users from executing pages out of the intended sequence. This does not eliminate the need for parameter validation. However, it does eliminate a major development hassle by giving the developer positive control of the sequence in which pages are visited and actions are executed.

Limitations

The flow controller presented here is intentionally simplified and suffers from the following three key limitations that will be addressed in future articles:

1. It provides no way to enable the back button. When a user goes back to a previous “page,” the browser retrieves the earlier page from cache unless you’ve turned off caching with one of the HTTP headers. Because the browser doesn’t send a request to the server, your application is unaware that a transition to a previous state has taken place and will


not be expecting the events from the previous state. Thus, use of the back button results in an error message. The refresh button suffers from the same problem. In a future article, we’ll look at how to make the flow controller smart enough to handle these (and double submits) gracefully.

2. For code brevity, the example code uses Session variables, ignoring for now any locking issues. These are important, especially in pre-MX versions and when considering double submits. The subject will have to be addressed in a future article.
3. Likewise, Session variables don’t work across multiple CF servers in a Web farm. If this is critical, you can run CFMX Enterprise using J2EE Session variables in a J2EE app server that supports automatic mirroring of the Session scope. Pre-MX users can download <CF_MirrorState> from www.turbomanage.com. This tag lets you choose between Session and Client variables at runtime and removes the need to lock variables throughout your code.

Conclusion

The event-driven state machine is a good fit for complex Web applications that require decoupled flow of control between views because it models the reality that modern Web applications are not stateless like the underlying Web protocols. Refactoring your code to utilize a Web state machine will help you to write more robust, secure, and reusable code.

For Further Reading

- *A State Machine Engine for Web MVC:* www.uidesign.net/Articles/Papers/AStateMachineEngineforWeb.html
- *Open Source State Machines for User Interfaces Written in Java:* www.manageability.org/blog/stuff/open-source-statemachine-for-user-interfaces-written-in-java/view 

About the Author

David Chandler is a senior ColdFusion developer with Digital Insight in Atlanta, GA and the author of Running a Perfect Web Site (Que, 1995). Chandler is the creator of Turbomanage, a metadata-driven framework for ColdFusion applications, and holds a patent on a method of organizing hierarchical data in a relational database.

dchandler@turbomanage.com

Listing 1

```
<!---
    flow_map.cfm

    Outer switch looks at current state
    Inner switch looks at current event
--->

<!--- Include common variables or functions needed by model
and view code in this flow --->
<!--- libQuery.cfm contains query UDFs from cflib.org --->
<cfinclude template="util/libQuery.cfm">

<!--- BEGIN state transition map --->
<cfswitch expression="#Variables.currentState#">
```

```
<!--- In init state --->
<cfcase value="init">
    <cfinclude template="model/initflow.cfm">
    <cfswitch expression="#Variables.thisEvent#">
        <cfcase value="startFlow">
            <cfset Variables.nextState = "choose_customer">
        </cfcase>
    </cfswitch>
</cfcase>

<!--- In choose_customer state --->
<cfcase value="choose_customer">
    <cfswitch expression="#Variables.thisEvent#">
        <cfcase value="next">
            <cfinclude template="model/act_saveCustomerSelection.cfm">
        </cfcase>
    </cfswitch>
</cfcase>
```



```

        <cfset Variables.nextState = "choose_products">
    </cfcase>
</cfswitch>
</cfcase>

<!-- In choose products state --->
<cfcase value="choose_products">
    <cfinclude template="model/act_saveProductSelection.cfm">
    <cfswitch expression="#Variables.thisEvent#">
        <cfcase value="selectProduct">
            <cfset Variables.nextState = "choose_products">
        </cfcase>
        <cfcase value="finish">
            <cfset Variables.nextState = "review_order">
        </cfcase>
        <cfcase value="changeCustomer">
            <cfset Variables.nextState = "choose_customer">
        </cfcase>
    </cfswitch>
</cfcase>
</cfswitch>
<!-- END state transition map --->

```

Listing 2

```

<!--
stateful_flow_controller.cfm
David Chandler, 12/2/03

This file implements a simple state machine for application flow
control. On first entry, it sets up a data structure in the session
space that will be used to hold data from the successive forms as
the user moves through the flow.

The controller keeps track of the flow state, beginning in the init
state.
Each button or hyperlink in the flow is assigned an event name.
The controller looks for the event in the URL and moves to the next
state based on the combination of previous state and event.
--->

<!-- Set default event --->
<cfif StructKeyExists (URL, "event")>
    <cfset Variables.thisEvent = URL.event>
<cfelseif StructKeyExists (Form, "event")>
    <cfset Variables.thisEvent = Form.event>
<cfelse>
    <cfset Variables.thisEvent = "">
</cfif>

<cfif NOT StructKeyExists (Session, "sFlowData") OR Variables.thisEvent
is "startFlow">
    <!-- If structure not defined, initialize the flow --->
    <cfset Session.sFlowData = StructNew ()>
    <cfset Session.sFlowData.currentState = "init">
    <cfset Variables.thisEvent = "startFlow">
    <!-- Create space for user data that belongs to this flow --->
    <cfset Session.sUserData = StructNew ()>
</cfif>

<!-- Create reference to struct that holds user's data --->
<cfset thisFlow = Session.sUserData>

<!-- Get currentState variable for flow map --->
<cfset Variables.currentState = Session.sFlowData.currentState>

<cfif Len (Variables.thisEvent)>
    <!-- Include flow map --->
    <cfinclude template="flow_map.cfm">
    <!-- If nextState is valid --->
    <cfparam name="Variables.nextState" default="">
    <cfif Len (Variables.nextState)>
        <!-- Advance the state machine --->
        <cfset Session.sFlowData.currentState = Variables.nextState>
    <cfelse>
        <!-- currentState and event pair are not valid --->
        <cfoutput>Event #Variables.thisEvent# is invalid in state
#Session.sFlowData.currentState#.</cfoutput>
        <cfabort>
    </cfif>
</cfif>

<!-- include view code for current state --->
<cfinclude template="views/#Session.sFlowData.currentState#.cfm">

```

Listing 3

```

<CFQUERY NAME="qCustomers" DATASOURCE="Northwind">
SELECT      CustomerID, CompanyName
FROM        dbo.Customers
</CFQUERY>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
<head>
    <title>Choose Customer</title>
    <SCRIPT src="./typeahead.js"></SCRIPT>
</head>

<body>
<cfmodule template="../showMsg.cfm">

<form action="flowctl.cfm" method="post" id="mainForm">
<select name="customerID" size="1" id="customerID">
<option value="">Select a customer...
<cfoutput query="qCustomers">
    <option value="#qCustomers.CustomerID#">#qCustomers.CompanyName#
</cfoutput>
</select>
<input type="Submit" name="event" value="next">
</form>

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
mainForm.customerID.onkeydown = typeAhead;
//-->
</SCRIPT>

</body>
</html>

```

Listing 4

```

<!-- Init array of selected products --->
<cfparam name="thisFlow.sOrder.aProductSelections" default="#ArrayNew
(1)#">

<CFQUERY NAME="qProducts" DATASOURCE="Northwind">
SELECT      ProductID, ProductName
FROM        dbo.Products
</CFQUERY>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
<head>
    <title>Select Products</title>
</head>

<body>
<cfmodule template="../showMsg.cfm">
<a href="flowctl.cfm">Start over</a>

<p>
<form action="flowctl.cfm" method="post" id="mainForm">
<cfoutput>
You have currently selected the following customer:<br>
<cfdump var="#thisFlow.sOrder.sCust#">
<input type="Submit" name="event" value="changeCustomer">
</cfoutput>
<p></p>

<select name="productID" size="1" id="productID">
<option value="">Select a product to add...
<cfoutput query="qProducts">
    <option value="#qProducts.ProductID#">#qProducts.ProductName#
</cfoutput>
</select>
<input type="Submit" name="event" value="selectProduct">
</p>

You have currently selected the following products:<br>
<cfdump var="#thisFlow.sOrder.aProductSelections#">
<p>
<input type="Submit" name="event" value="finish">
</form>

</body>
</html>

```

Download the Code...
Go to www.coldfusionjournal.com

Creating and Using User-Defined Functions

A step-by-step guide

An old friend, whom I hadn't seen in over eight years, recently brought an eBay auction to my attention. After some rough times, someone was selling everything he owned so he could start his life completely from scratch. The seller was located in my home state (Connecticut), was a programmer computer-geek, and had long hair. "Jeff, is this you?" was the body of the e-mail.

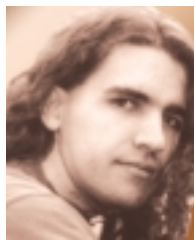
Of course, it wasn't, but the auction reminded me that sometimes it's better to start from scratch than to try to fix what you have.

I was thinking of rebuilding some of my older sites. These sites were built while CF5 was still in development and made very little use of user-defined functions (UDFs). ColdFusion Components were only a pipe dream at that time. Given the advent of these two features, it would be more efficient for me to rebuild these sites by starting from scratch than it would be to try to continue to modify the code base as is. This month I'm going to talk about user-defined functions, and next month I'll go into ColdFusion Components.

What Is a User-Defined Function?

Before going further, you'll need to understand what a user-defined function is. In simplest terms, a UDF is a way to assign a name to a chunk of code. When you use that name, ColdFusion will know to execute its associated code. Functions can accept arguments, and they should always return a value (although it is possible to write a UDF that doesn't return anything). UDFs operate in the same manner as built-in functions, except that you must provide the code behind the function.

If you find that you are using a similar block of code many times over in your application(s), then you will probably benefit from putting that code in a function. ColdFusion provides many different string processing functions, but one I was surprised not to find is a ProperCase function. A ProperCase



By Jeffry Houser

function is one that will capitalize the first letter of each word in a string, and lowercase all other letters. Often when displaying data, such as someone's name, you'll want to make sure that it is formatted properly before displaying it. That's when functions like ProperCase come in handy. I'm going to teach you how to create your own UDFs by creating a proper case function.

The ProperCase Algorithm

Before you can create a function you must first understand the block of code, or algorithm, that will be executed whenever the function is called. The requirements for the proper case function are:

- Each letter that starts a word is turned capital.
- A letter starts a word if the character before it is a space.
- A letter starts a word if there is no character before it.
- Each letter that is not the first letter of a word should be put into lowercase.

To force a character into capital or lowercase, we can use ColdFusion's built-in functions, UCase and LCase, respectively. Each one accepts a single parameter of the string to be moved upper or lower. There are many different ways to approach this coding. In our code we'll need to keep track of the current letter of the string and the previous letter of the string. We can compare the previous, or current letter, with a space or empty string to see if the letter should be put into upper or lower case. Another option might be to use an lcase on the whole string, and then loop through the string as a space delimited list. I am not using this method because ColdFusion ignores empty list elements, and our resulting string structure may not be preserved. See Listing 1 to see how we implement the code, as if it were in a normal CF page.

The code starts by setting the string we want to put in ProperCase. In normal circumstances, this string would be something we took out of a database or that was input by a user. Then we initialize variables for the current character, the previous character, and the result string. Then the code enters an index loop. The code loops from the first character in the string (character 1) to the last character of the string. The last character is determined by using the Len function on the string. Len is short for length, and it returns the number of characters in the string.

For each character in the string, we want to compare it with the previous character to decide if it goes into upper or lowercase. The first two lines in the loop set the previous value to the current value. This sounds strange, but it's what we want to do. The current variable still holds the same value as the most recent iteration of the loop, which means it is our previous value. Then the code updates the current value using the mid function. The mid function selects a number of characters out of the middle of the string. It accepts three values: the string to remove characters from, the first character we want to take, and the length of the resultant string we want. The code is pulling characters out of our string. It starts at the current character of the string, which we can tell by using the counter variable. The length is 1 because we are comparing single characters.

The next part of the loop decides what to do with the current character. If the previous character is a space or an empty string and the current character is not a space or an empty string, then

we move it to uppercase and append it to the result. This prevents us from putting empty spaces into uppercase. If the current character is not a space nor empty string, then it is put into lowercase. Otherwise we just append the character to the result string. After the loop there is additional code to output the original value and the proper case value.

The Tags Used to Create Functions

You don't want to have to write this chunk of code every time you need to put a string into proper case, do you? I didn't think so. Let's create a UDF based on this algorithm. There are two different ways to create UDFs in ColdFusion. The first is through the use of the cffunction, cfargument, and cfreturn tags. The second is to use CFScript. In ColdFusion 5, UDFs could only be created in CFScript, but ColdFusion MX introduced the tag-based syntax, which we're going to start with. It's also easier to understand for people who haven't been exposed to CFScript.

The cffunction tag is used to define

the function. It accepts the following attributes:

- **Name:** The name attribute specifies the name of the function. Function naming is similar to variable naming. The names of built-in CFML functions are not allowed. This is a required attribute.
- **Returntype:** The returntype specifies the return type of the function. The valid values are Any, Array, Binary, Boolean, Date, guid, numeric, query, string, struct, uuid, variable name, void, or the name of a ColdFusion Component. The default is "Any", which means that anything can be returned from the function. This attribute is optional.
- **Roles:** The roles attribute is used to specify the roles that are allowed to access this function. This refers to the roles that are set when a user logs in using the ColdFusion cfloginuser tag. It applies only inside a ColdFusion Component and doesn't apply to standalone functions (UDFs not inside a CFC). This attribute is optional.
- **Access:** This attribute applies only to functions inside ColdFusion



TM

HostMySite.com

Built for ColdFusion Pros by ColdFusion Pros

- 24 / 7 / 365 Phone Support
- 99.9+% Uptime
- Macromedia Alliance Partner
- "Full Control" Panel
- CFMX 6.1 or CF 5.0
- SQL Server 2000 or 7.0
- Custom Tags Welcome

plans from

\$8.95 / mo.

FREE Domain Name*

FREE Setup

FREE 2 Months

Visit www.HostMySite.com/cfdj for:

2 Months Free

FREE Setup and FREE Domain Name

*on any annual shared hosting plan

call today

877•248•HOST
(4678)

Components and specifies how the function exposes itself outside the CFC. The allowed values are: `private`, which means only code inside the component can execute this function; `package`, which means only code inside the component package can execute it; `public`, which means anyone can execute it; and `remote` which means it can be evoked remotely via Web services or Flash Remoting or a URL interface. This attribute is optional and defaults to `public`.

- **Output:** Specifies whether output will be displayed inside the component or not. If set to `yes`, then the body of the function will be treated as if it were inside a `cfoutput`. If set to `no`, then the body of the function will be treated as if it were inside a `cfsilent`. This attribute is optional and defaults to `yes`.
- **Displayname:** Displayname is an optional attribute used only for display in the metadata generated for a component. It has no effect on functionality.
- **Hint:** The hint attribute is used to display comments or additional documentation in the component metadata. It has no effect on functionality and is optional.

The `cfargument` tag is used to define arguments that can be passed into the function. It must always be used inside a `cffunction` tag block body immediately following the opening `cffunction` tag. These are its arguments:

- **Name:** The name attribute specifies the name of the argument and is required.
- **Type:** The argument type is similar to the `returntype` of the `cffunction` tag. It specifies the type of value that is expected for this argument. The default is `any`, which means that any valid value is acceptable. Other values are `Array`, `Binary`, `Boolean`, `Date`, `guid`, `numeric`, `query`, `string`, `struct`, `uuid`, `variablename`, `void`, or the name of a ColdFusion Component.
- **Required:** This argument is either `true` or `false`, and it specifies whether the argument is required or not. If a required value is not passed to the function, then ColdFusion will throw an error. This is an optional argument and the default is `false`, or not required.
- **Default:** The default attribute specifies the value that this argument will be given if it is not specified. This is mutually exclusive with the required attribute.
- **Displayname and Hint:** The displayname and hint attributes are used strictly for metadata (documentation) purposes in components, just like these attributes are used with the `cffunction` name, and are not required.

The last tag that can be used in relation to functions is the `cfreturn` tag. The `cfreturn` tag doesn't have any attributes. It operates in a way similar to the `cfif` tag – you simply place an expression in the tag (there is no closing `cfreturn` tag). This expression is the return value.

Creating Your First Function


With an understanding of the tags that are used to create user-defined functions, you can now examine the `ProperCase` code turned into a function (see Listing 2).

The function starts with the `cffunction` tag. This is a fairly simple function declaration specifying only that there is no output inside the function, it will return a string, and the name of the function is `ProperCase`. Then one argument is defined, `str`, which is the string that the function will put into proper case. The argument value passed is available in the arguments scope.

The next step in the function declares our local variables. Local variables in functions are private to the function – this means that they are protected from variables with the same name in other pages and functions and that they don't expose themselves outside of the function. Private variables are put in an unnamed scope, so you only access it by the variable name with no scope specified. Local variables must be declared at the beginning of the function immediately following any arguments. They are done using the `cfset` tag with the `var` keyword. As a best practice, be sure to `var` all variables used in the function that you do not want to exist outside of the function. This includes the counter variable, which we use in the loop. The current character, previous character, and result string values are all initialized.

The meat of the code remains almost identical to our initial code. In fact I copied and pasted it from the CF page inside the function page. I made one slight change, to specify the argument scope whenever the `str` variable is referenced. The last line of the function uses the `cfreturn` tag to return the result string. If you test the code you'll find it runs no differently than the brute force code approach, but is now easily executed from any part of my files – reusability achieved!

Conclusion

The most common way to call a user-defined function is to do so the same way as a built-in function. You can call it within any ColdFusion expression. There is an alternate way, including using the `cfinvoke` tag, if the function is part of a ColdFusion Component. Additionally, there is a CFScript syntax for writing your own functions. You can read more about that at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/udfs5.htm>. Before you go about writing your own function, you may want to check to see if it exists at the common function library project, found at www.cflib.org. `cflib` is a great learning and development resource and you'll find many functions to do many different things. Next month I'm going to take a look at ColdFusion Components. While functions are great for modularizing functionality, components allow you to modularize both data and functionality. 

About the Author

Jeffrey Houser has been working with computers for over 20 years and in Web development for over 8 years. He owns a consulting company, and has authored three separate books on CF; most recently ColdFusion MX: The Complete Reference (McGraw-Hill Osborne Media).

jeff@instantcoldfusion.com

Listing 1

```
<cfset str = "jeFFry h0user.">

<cfset Current = "">
<cfset Previous = "">
<cfset Result = "">

<cfloop index="Counter" from="1" to="#Len(str)#">
  <cfset Previous = Current>
  <cfset Current = mid(str,counter,1)>

  <cfif ((Previous is " ") or (Previous is "")) and
    ((Current is not " ") or (Current is not ""))>
    <cfset result = result & UCase(Current)>
  <cfelseif ((Current is not " ") or (Current is not ""))>
    <cfset result = result & #LCase(current)#>
  <cfelse>
    <cfset result = result & current>
  </cfif>
</cfloop>
```

Listing 2

```
<cffunction name="ProperCase" returntype="string" output="No">
  <cfargument name="str" required="Yes" type="string">
  <cfset var Current = "">
  <cfset var Previous = "">
  <cfset var Result = "">
  <cfset var counter = 0>

  <cfloop index="Counter" from="1" to="#Len(arguments.str)#">
    <cfset Previous = Current>
    <cfset Current = mid(str,counter,1)>
```

```
<cfif ((Previous is " ") or (Previous is "")) and
  ((Current is not " ") or (Current is not ""))>
  <cfset result = result & UCase(Current)>
<cfelseif ((Current is not " ") or (Current is not ""))>
  <cfset result = result & #LCase(current)#>
<cfelse>
  <cfset result = result & current>
</cfif>
</cfloop>

<cfreturn result>
</cffunction>
```

Download the Code...
Go to www.coldfusionjournal.com

2004 RCAs Readers' Choice Awards Nominations Open

"The Oscars of the Software Industry"



Nominate the Best of the Best:
Tell us what tools, solutions, services,
and books deserve recognition.

Cast your vote at
www.SYS-CON.com

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
2004 READERS' CHOICE AWARDS	WWW.SYS-CON.COM	888-303-5282	31
ACTIVEPDF	WWW.ACTIVEPDF.COM		15
CF_UNDERGROUND VI	WWW.CFCONF.ORG/CF_UNDERGROUND6/	301-424-3903	21
CFDYNAMICS	WWW.CFDYNAMICS.COM	866-233-9626	11
CLEARNOVA	WWW.CLEARNOVA.COM/thinkcap		25
CFDJ	WWW.SYS-CON.COM/COLDFUSION/	888-303-5282	45
FUSETALK	WWW.FUSETALK.COM	866-477-7542	33
HAL HELMS, INC	WWW.HALHELMS.COM		35
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM/CFDJ	877-248-4678	29
ISSJ	WWW.ISSJOURNAL.COM	888-303-5282	49
INTERAKT ONLINE	WWW.INTERAKTONLINE.COM		6
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800-379-7729	COVER IV
MACROMEDIA	WWW.MACROMEDIA.COM/GO/VOLVO		COVER II & Page 3
MACROMEDIA MAX 2004	WWW.MACROMEDIA.COM/go/max		13
MX DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM	888-303-5282	49
SEAPINE SOFTWARE	WWW.SEAPINE.COM	888-683-6456	19
SECRETS OF THE CF MASTERS	WWW.SYS-CON.COM/FREECD	888-303-5282	39
SERVERSIDE	WWW.SERVERSIDE.NET	888-682-2544	4
WEB SERVICES EDGE EAST	WWW.SYS-CON.COM/EDGE	201-802-3066	43
WEBCORE TECH	WWW.WEBCORETECH.COM	877-WCT-HOST	COVER III

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in ColdFusion Developer's Journal. Advertisements are to be printed at the discretion of the Publisher. This disclaimer includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions. This index is provided as an additional service to our readers.

Getting Ready for MAX

— continued from page 5

Also this month, Jeffry Houser explores one of the most important programming constructs in CFML, User Defined Functions, in his **CF-101** column.

This month also features a couple of articles about methodology and design. Hal Helms offers a good overview of extreme programming and his thoughts on the methodology. There's also an article about developing leadership skills by Reuben Poon. On the design front, David Chandler has written an article about controlling page loads using what he likes to call a "state machine" framework. Also addressing how to architect applications, I contributed an article this month that shows (using a case study example) how I used J2EE design patterns to enhance functionality and performance in a large enterprise application framework.

As always, I am open to ideas for articles, issue focus topics, magazine format, or any other way to make *CFDJ* the best resource for ColdFusion developers. If you have any ideas and are at MAX, be sure to grab me and let me know your thoughts about the magazine. If you don't grab me at MAX, be sure to send me an e-mail... there's no point in having good ideas if you keep them to yourself. I'm really looking forward to the conference and to seeing the articles and ideas produced by MAX in next month's issue. I hope that you are, too.



CFC Data Exchange

Open up the doors to a whole new realm of development solutions using CFML



Beginning with the release of ColdFusion MX, the ColdFusion Application Server now runs on top of an underlying Java engine. Java, as we all know, is an object-oriented language. However, the question of whether CFML is an object-oriented language has been under debate since the ColdFusion Component (CFC) Framework was introduced in ColdFusion MX.

Too much energy has been spent bantering about whether CFML is object oriented, rather than exploring uses for the object-oriented features that CFCs brought to the language. Among these features were inheritance, support for the creation of Web services, the ability to encapsulate data and busi-



By Simon Horwith

ness logic in objects and to perform instance-based development, and metadata.

So many new features mean that the rules have changed for ColdFusion developers. We must now relearn and redefine best practices, which not only means changing the way we write code but also the way that we plan our applications. Fortunately, we can take advantage of a wealth of knowledge about what does and doesn't work. We can do this by studying the design patterns and development techniques that have been tried and tested by object-oriented

developers, most notably Java developers, for more than a decade. Because of the very nature of Web-application development, J2EE design patterns are particularly applicable when architecting applications with ColdFusion.

Unfortunately, understanding design patterns is only the first step toward building CFML applications the right way. There still remains the challenge of not only selecting the right pattern for the job, but also knowing how to implement a pattern (or not) in your code.

No single article, or book for that matter, could ever hope to

provide the magic answer to determining what pattern to use when, or even how to properly implement any one design pattern in CFML. There are some patterns that lend themselves to the language very well and others that do not, and most could be implemented in any one of several manners. It would take several months, or years even, to properly test and document it all. The purpose of this article is to shed some light on a general realm of design patterns. This article will also case study a recent implementation that allowed me to quickly help reduce overhead, boost performance, and add functionality to a client application. It is just one example of how knowledge of design patterns and creativity with CFML can allow you to achieve your goals quickly and efficiently.

Working as a consultant, I am often asked to advise on architectural decisions as well as troubleshoot existing architectures. A client of mine developed an application (a framework) with ColdFusion allowing rather inexperienced developers to quickly and easily create data entry and data display screens on the Web. The idea being that defining the database table with XML and adding a few lines of code to a display file and a CFC (which inherits from a parent "object" code base) is all it takes to create screens that have a uniform look and feel and reproduce the screens in a massive legacy application. The client hopes to move the entire legacy application onto the Web within two years.

My role for the past year has primarily been to assess risks, advise on the best course of action, and to come in and fix problems when all else has failed. As you can imagine, at any given moment I might be researching several issues at once.

Just prior to a recent major release of the software, I was asked to look into two major issues. One was the replication of sessions when servers in our J2EE clusters failover and the other was application performance in general.

The first thing I did was tackle the session replication problem. As it turns out, this was right around the time that J-Run Updater 4 and the ColdFusion MX 6.1 Updater were both released to the public. Unfortunately even with both updaters installed, after all of my testing I was unable to consistently make ColdFusion session variables replicate to a server that was added to the cluster. Session replication and failover works pretty consistently – when you turn off a server everything fails over and replicates fine. However, when you add a server back into the cluster, the session doesn't always replicate onto the new server properly. The application server console always reports a J2EESessionScope error when a server rejoins the cluster. Sometimes ignoring this message appears to be okay, and sometimes it does not. It's also worth noting that sending requests while a server was in the process of rejoining the cluster seems to encourage session replication failure. All of my tests with "pure" Java, even simple JSP files, consistently worked. Sessions always replicated just fine. One of my rules of thumb is to never spend too much time on one error, so I put that problem on the back burner and focused on the other problem: performance. We'll revisit the question of session replication later.

There are a lot of things that can kill performance and a lot of things that can be done to speed things up in an applica-

Web based collaboration made to order.

Collaboration

Flexible Security, 508 Compliance,
Offline Capabilities, Reporting, Easy
Integration and much more...

1-866-477-7542



2003 CFDJ Readers' Choice Award
Winner for Best eBusiness Software
& Best Web Application

Discussion Forums

www.fusetalk.com



Discussion forum solutions that make web-based collaboration risk-free and easy.

tion. Every developer has a mental list of the things they do on a page or in an application to make it efficient (use CFSCRIPT, avoid unnecessary looping/conditional logic/HTTP, cache things in memory, optimize and minimize database calls, etc.). The list can be huge but the last two points, taking advantage of the server's memory and minimizing/optimizing calls to the database, are the two most common ways developers speed up their applications. But what can you do to speed up an application that already has all of these things in place? Go line by line through the code looking for small tweaks here and there?

You could do that, but the performance gains would be minimal and it would take a long time in this case. The application I was working with already consisted of about 600 files including not only the framework, but also the screen definitions and accompanying logic as well. The application I was asked to optimize also already had a great deal of caching and code reuse in place, and all of the database access had been gone over with a fine-tooth comb. When faced with a situation like this, I find that it's a good strategy to take a step back and look at the big picture.

Looking at the entire application from a distance must be a lot like looking at Earth from outer space. There are no countries, no borders, and no politics. There's just water, land, and clouds all working together to create that delicate balance needed to sustain life. All of the minute details blur together, allowing you to concentrate on the system as a whole rather than trying to focus on all of the smaller parts, which in the big picture are insignificant. When I thought about how I would model this client's application and how all of the parts of the model interacted, I looked for problems. If I didn't find any, I'd take a step back. Eventually, a potential problem almost always presents itself. In this case, when I finally got to the point of blurring modules into single tiers that interact with one other, something dawned on me: tier communication patterns.

It had been some time since I'd thought about all of the design patterns I've studied in the past. But I remembered that there are a few patterns I'd read about that deal primarily with communication between tiers in an application, whether those tiers were the presentation and business logic tiers or business logic and database. The application I was optimizing had a terribly high number of CFC instances being passed around between the presentation layer and the business components, between Java and ColdFusion, and between business components and other business components. I hadn't thought about this as a problem because the individual CFC instantiations and method calls were taking very little time at all. Most of the CFCs were instantiated and cached in memory once and then reused throughout the application. Still, there is a certain amount of overhead in passing instances of complex objects around in memory as well as in persisting all of the instruction sets for dozens of components. Many of these were even cached in sessions (for justifiable reasons). A few patterns came to mind that might shed light on a possible solution: Data Transfer Objects, Lazy Load Pattern, and the IsDirty Pattern. So I went back to the books and brushed up on the intricacies of these three pat-

terns and some of their related patterns.

After refreshing my memory about these patterns, it became clear that what I was looking for was something along the lines of the Data Transfer Hash Pattern (the other patterns I mentioned have qualities I'm looking for but are more specific to database interaction). The idea behind the Data Transfer Hash Pattern is that passing instances of objects around carries with it a lot of overhead, so hashmap representations are used instead. Think of a hashmap as the Java equivalent of a ColdFusion structure. It's much more lightweight moving a hashmap data structure around rather than a complex object that contains functionality (e.g., a CFC instance). The one downside is that there's no easy way to enforce the values within that hashmap. This is a problem more easily remedied in Java by using Objects, but all too familiar and not-so-easily remedied in ColdFusion, as our only option is to use a CFC with a constructor. The IsDirty Pattern, in a nutshell, is all about setting a Boolean flag in a data container so that database commands to manipulate data are executed only when data has actually changed. The Lazy Load Pattern is similar to IsDirty in that it reduces database connectivity overhead. However, it does so by not retrieving data until it's specifically requested. In implementing the Data Transfer Hash Pattern, I will also borrow from these ideas in that I will keep external object access to a minimum until it is required. But the solution is still in essence more of a Data Transfer Hash Pattern solution.

So how do I put this idea of only transferring simple data collections rather than object instances into practice? Ideally, I'd like a way to represent the data contained within a CFC instance as a structure. Furthermore, I want to do so in a way that's flexible, easy for the end users of the framework to implement, and doesn't break the existing code base. Fortunately, there are some features in CFML that I can take advantage of to make this work.

For one thing, we have inheritance. In my particular scenario I could easily modify `component.cfc` to offer this functionality within all CFCs. I described this technique in an article titled "component.cfc: The Mother of All Components" in the November 2003 issue of *ColdFusion Developer's Journal*. The code has to be very portable, which means that I don't want to rely on this technique. Fortunately for me, because my "fix" must be implemented in the context of a framework and all CFCs in that framework inherit from a "master" CFC, I can easily create a component and specify that the "parent" inherits from it (via the `<CFCOMPONENT>` tag "extends" attribute). It also occurred to me that until an elegant solution to my session replication woes is found, this pattern offers the added bonus of helping to address that problem as well. In ColdFusion MX (6.1 or prior) CFC instances are not serializable. This means that they cannot be replicated across servers in a cluster. I knew this and did advise my client that all CFC instances in the session scope had to be made nonpersistent. This resulted in two code bases. One we wanted to implement but couldn't until serialized CFC instances is supported (a feature rumored to be in Blackstone). The other one works around the issue by not keeping CFC instances in the session scope. If I could present a way to represent CFC instances as

structures, not only would it reduce overhead, but it would also offer a means to persist CFCs and have them replicate. This could be accomplished by serializing and deserializing them as structures. Although I was having trouble making serialization work for ColdFusion, structures map to hashmaps (and vice versa) quite well. Plus I'd already proven that Java session information serializes just fine.


My solution took advantage of the metadata created for CFCs: metadata exposed with the `getMetaData()` function. I created three methods. One simply accepts a CFC instance and returns its metadata in a more "friendly" format. Another accepts a structure and the name (including full package) of a component and returns an instance of that component populated with the structure data. The third method accepts a CFC instance and returns a structure containing all of its properties. How could I guarantee that I retrieved all of the properties?

My requirements were simple, and fortunately the framework coding standards that were already in place assured me that the following conditions already exist. When serializing the data in a CFC within a structure, I look for any property defined with a `<CFPROPERTY>` tag. If a method exists with the name "get" plus the property name, the code calls that method to get the value. If a property is defined that doesn't have a "getter," I copy it directly from the component's public "this" scope if it exists. When creating a component instance from a structure, I create an instance of the component and look for an `init()` method. If a non-private `init()` method exists and the structure contains keys with names of any and all required parameters, I pass the parameters to the `init()` method. After calling the `init()` method, if it exists, the code then loops over the structure that was passed and calls any method named "set" plus the structure key name (passing it the structure key value). If no "setter" exists but a public property with the same name as the current key name exists, the code sets a variable in the component's "this" scope with the name-value pair from the structure. It's probably not the most robust or flexible solution in the world, but it does serve my purpose.

The rest of the implementation details are just a matter of persisting structures whenever possible if an object isn't likely to receive much activity and serializing that structure as a CFC instance in the event that an actual object is required in a certain scenario. It's worth noting that in a system in which the CFC instances receive frequent method calls that manipulate their data, the overhead serializing and deserializing to and from structures carries more performance hit than simply keeping a component instance in memory. However, often in other scenarios you can free up a significant amount of memory and other resources by not keeping so many objects in memory. This may result in better performance (it did in my scenario).

More and more it is becoming evident that it's necessary to have at least a basic understanding of design patterns. Coupled with the features in ColdFusion MX, this opens up the doors to a whole new realm of development solutions using CFML. Any of you who are interested in seeing the code I used to implement this solution, may download it

from www.sys-con.com/coldfusion/sourceec.cfm or from www.horwith.com/downloads/dto.zip.

I encourage you to not just examine my code but (more important) think about alternative approaches to problems and creative methods for implementing the design patterns that already exist and are just waiting to be explored by ColdFusion developers. It's also worth noting that the technique I've outlined is what J2EE application servers already do to serialize sessions in a cluster. The application server itself is a very good place to look for inspiration when trying to determine how best to meet business requirements in your applications. 

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. He is a freelance software architect currently consulting with companies in London, England. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and consults on ColdFusion applications that leverage Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com

simon@horwith.com

"I was totally intimidated by Java, but I knew I had to learn it. Your class taught me what I honestly thought I couldn't be taught." - Sharon T

Java for ColdFusion Programmers?

Java for ColdFusion Programmers, the five-day Hal Helms, Inc. training class is designed for ColdFusion programmers; no previous Java experience is needed. You'll learn how to think in objects and program in Java.

For class information and registration, come to halhelms.com.



HTTP Status Codes: Do the Unthinkable

Implement threading using CFML

HTTTP status codes can help you implement threading and more. Here are a couple of ideas.

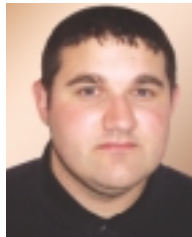
Although the power of ColdFusion allows us as developers to do many things very quickly compared to many other languages, there are times when we find CFML does not offer all of the functionality required to accomplish a task. It can be quite frustrating to find that your end result of many hours of programming is limited by your tools, rather than your skills or experience. In this article, I will show you how to implement a couple of interesting and relatively simple solutions to problems I've come across in the past, through the use of an oftentimes underutilized tool – HTTP status codes.

Have you visited Web sites that had a simple HTML form for voting that was able to tally your vote without refreshing the page? On the surface, it would seem that this was accomplished through JavaScript, but it's actually done by HTTP status codes. Using our sample application, the ColdFusion MX Hotfix Center, I will show you how you can do this yourself. I've also implemented another example into the sample application showing how to leverage status codes to implement a scaled-down version of threading using ColdFusion MX.

Admittedly, the specific techniques used in this article are perhaps not applicable to all typical Web applications. For instance, the first example could also be done in Flash, but I included it just as an example of what you could potentially use HTTP status codes to do. However, I have found the use of HTTP status codes very useful when working in complex, multi-tiered environments with slow third-party resources that a ColdFusion application is dependent on. A specific example of this is a situation I've worked on in the past: a ColdFusion application that connected to a slow legacy database system for reporting. Because the legacy database system took tens of seconds instead of milliseconds to return queries, it was very useful to run each query simultaneously using HTTP status codes – instead of one at a time – to vastly speed up the application.

The HTTP Protocol

The HTTP protocol is a completely separate subject of its own, so I'll just briefly brush over it in case you are not famil-



By Brandon Harper

iar with its general premise. When a client (such as a Web browser) makes a request to a Web server, HTTP status codes are returned to the client in the header of the Web server HTTP response. The client uses these codes to decide which action it should perform. For instance, if the client receives a code of "200", which is "OK", it knows the page was found and has been returned in the body of the response. In the examples that follow, we will concentrate on a specific status code: status code 204. If you'd like to read more about the HTTP protocol, please

check the Resources section at the end of this article for links that describe it in detail.

HTTP status code 204 essentially tells the client to not change what it's displaying, as there is no content coming back from the server. What does this mean? In a nutshell, it allows users to submit information without having to wait for the page to reload, much as if they were using a Flash-based form. This minor detail is very powerful at the presentation level, in addition to providing us with a way to implement a scaled-down version of threading using ColdFusion MX.

Environment Setup

The ColdFusion MX Hotfix Center is a quick application I wrote to show a couple of simple possibilities using status code 204. To make it easier to install and use, I chose to create a simple XML file rather than a database, so you will need to make sure the ColdFusion MX 6.1 server you use for this sample has the <cffile> tag enabled, and that the server can access the Internet, as I also use the <cfhttp> tag to download hotfixes.

After confirming the aforementioned details on your ColdFusion server, I highly recommend downloading the sample code in this application from the **ColdFusion Developer's Journal** site (www.sys-con.com/coldfusion/sourcecfm) and deploying it into a subdirectory named "hotfix". From there, you will need to open the Application.cfm file and make sure the configuration lines shown in Listing 1 are set up correctly for your environment.

After you've saved your changes to this file to reflect your environment, you should be able to browse to the index.cfm file of the application via a Web browser, and it will return a page that looks similar to the one shown in Figure 1. If you see a page similar to this, you are ready to continue.

Voting Example

Often seen on news sites, blogs, and various other forums on the Web, voting is a powerful way for Web site owners to gather information from their visitors. It's very simple for visitors to participate, and they usually don't mind taking a few seconds to vote on an issue that is presented to them.

In our first bit of code contained within index.cfm (see Listing 2), we will implement a simple form that is dynamically generated by the XML mentioned earlier. This component then posts the vote information to be tallied back to our Hotfix component, hotfix.cfc.

As you can see, it's just a basic HTML form that posts to the hotfix.cfc method. We are also calling the small bit of inline JavaScript in the index.cfm file to disable the submit button in this form after it's been submitted so that a user is not able to vote more than once (see Listing 3).

Of course there are also a lot of other considerations to take into account when creating a voting system – such as how to avoid duplicate votes, for instance – but those issues are outside the scope of this article.

Listing 4 shows the important functions from hotfix.cfc that are involved with tallying the votes coming from this

form, as well as returning the status code 204 to our Web browser.

The first, tallyVote, is the function that accepts the form post and updates the Request scope to reflect the new vote. The second function shown in Listing 4, return204, which tallyVote calls, returns the status code 204 to our browser, and thus ends the request as far as our browser is concerned.

To test this functionality using your browser, cast a few votes, and then refresh the page in your browser. Notice how the vote tally has changed? Each time you submitted the form, the server responded to your browser with the 204 status code, and it did not try to refresh the page or load other content.

Not bad for a few lines of code, eh? This is obviously a very simple thing to do, but it's used so rarely on the Web that I wanted to show other developers the power of using it. The code in Listing 4 represents a pretty basic example of how to use status codes. We'll extend that idea in the next section.

Threading Overview

As a programmer with a traditional software development background, one of the things I repeatedly cite as a shortcoming of the ColdFusion language is the lack of threading support within

CFML. Granted, the ColdFusion MX server itself is multithreaded, as it handles simultaneous requests; however, there is no mechanism for spawning threads within the language itself.

According to recent blog entries and user presentations about Blackstone, the next version of ColdFusion MX will include threading support. Until then, we still have problems we need to solve – and finding a way to do threading was one of them – until now.

Threading, like the HTTP protocol, is a topic on which books have been written, so I will cover only a few basics here. Essentially, threading is a way of assigning tasks to parallel processes so they can be run at the same time, rather than serially, in which case one process or function must finish what it's doing before the next will be run. Traditionally, threading is most often used in languages like C++ or Java in GUI applications that need to spawn threads to perform operations in the background while the display of an application is doing something else, such as updating the user on the status of a thread that it spawned in the background. An example of this would be the status bar you are shown when copying a file in Windows. The file copying is a spawned thread running in the background, while the display updates itself as the files are copied.

In most Web applications, it's not quite as obvious where having threading functionality would be a big advantage. However, in enterprise-level applications, having threading at your disposal is more necessity than desire. A prime example would be a reporting application that has to run many queries, all of which take a long time to complete; firing all of these queries off at the same time in parallel would cut down the processing time significantly.

Implementing Threading in ColdFusion MX

Accomplishing threading in CFMX is not much different from our first code example. We still need to return the same 204 status code to our client; however, this time our client will actually be the <cfhttp> tag. Keep in mind that the performance drawbacks associated with making HTTP requests must always be weighed against the possible benefits of their use. That said, there are some

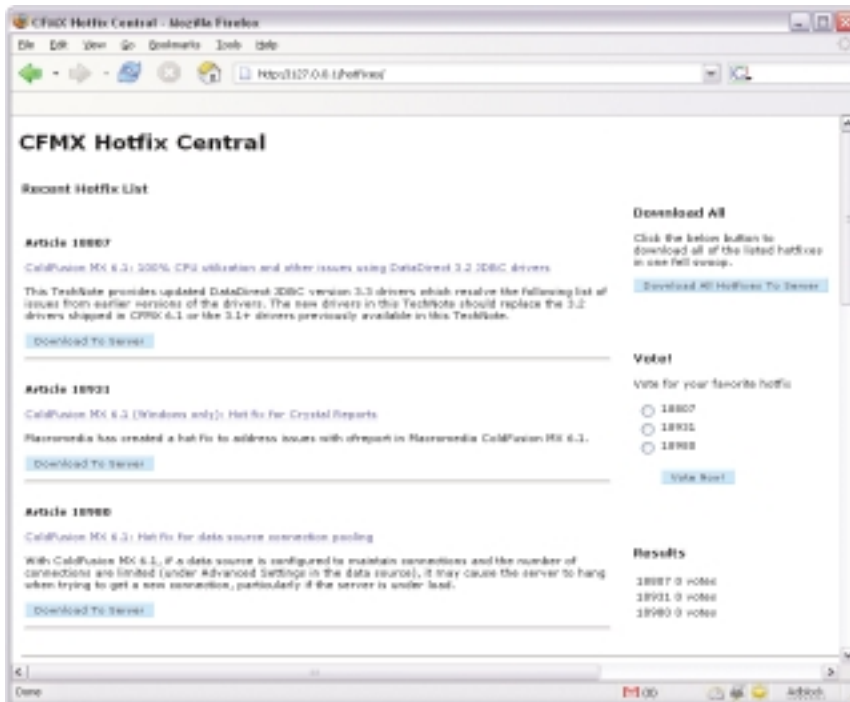


Figure 1: A screen shot of the application

interesting real-world uses for multiple simultaneous requests.

In this example we will use threading to download to the server all of the hotfixes contained in the hotfix.xml file. First, you will want to open up the index.cfm file from the source code and scroll down to take a look at the simple form that downloads all of the hotfixes at the same time (see Listing 5).

This form is posted to hotfix.cfc, calling the method threadDownloads, (see Listing 6), which is the one that actually spawns all of the new threads by looping through the various hotfix IDs and establishing a <cfhttp> call for each one to invoke a new thread, which downloads that particular hotfix to the server.

Caveats to These Techniques

Because the above example of threading is essentially akin to spawning off extra processes, extra planning is required to get them to work depending on how you decide to implement this technique. For many types of the operations that you may wish to thread, it will take quite a bit more effort than traditional threading, as each thread is actually a separately spawned process rather than something that is part of the process, including the same RAM and environment as the process which spawned the thread.

Given that each "thread" is really just a separate request, you will have to find a way for the results of the spawned thread to talk back to the process that spawned them, if they are supposed to be returning data. One of the ways I have currently implemented this is a threading system that uses its own internal status codes and stores the results of a given thread in a database using WDDX encoding.

Once the parent process queries the database and finds that a thread is done, it's able to get its stored results out of the database.

When using this technique you will need to make sure that you have debugging turned off. If any debugging or other output is returned to the client, it will actually wait for the request to finish as well as displaying whatever is returned in plain text

format, thereby defeating the purpose of using your own status codes.


Conclusion

Although these techniques are pretty simple to implement, either very few people are aware of them or very little has been written about them online. The samples I've given you for using HTTP status codes are pretty basic in nature, but I hope they will provide you as a ColdFusion developer with an additional tool to use. I definitely look forward to seeing how others use this technique in their Web applications, and hope that you find it as useful as I have.

Resources

- *RFC 2616 - HTTP 1.1:*
www.w3.org/Protocols/History.html#HTTP11
- *HTTP 1.1 Status Code Definitions:*
www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

Acknowledgments

I would like to thank Jake Sutton and Neal Enssle for contributing brainpower to figuring out these techniques. It was definitely a group effort to stumble upon this gem and find interesting ways to implement it. 

About the Author

Brandon Harper has been working with computers for over 20 years, has been developing for the Web for 10 years, and has been implementing solutions using ColdFusion since 1998. He is currently a software developer at InsightAmerica, a company that provides risk mitigation technology to Fortune 500 companies, small businesses, and individuals using analytically enhanced data. Brandon was also a technical editor for Inside ColdFusion MX (New Riders).

brandon@devnull.com

Listing 1:

```
<!-- BH: Change these variables to reflect the path of the app on your
webserver -->
<cfset Request.thisApp = StructNew() />
<!-- BH: CFC Path to this dir -->
<cfset Request.thisApp.CFCPath = 'hotfixes.' />
<!-- BH: File path to the dir this app resides in -->
<cfset Request.thisApp.FilePath = 'C:\wwwroot\Default\hotfixes\' />
<!-- BH: XML File which contains our app information -->
<cfset Request.thisApp.XMLFile = 'hotfixes.xml' />

<!-- BH: This is where our hotfix info will be stored -->
<cfset Request.hotfixes = StructNew() />

<!-- BH: Invoke the Hotfixes Application -->
<cfset Request.CFC = StructNew() />
<cfset Request.CFC.Hotfixes = CreateObject("component",
"#Request.thisApp.CFCPath#hotfixes") />
<cfset Request.CFC.Hotfixes.init() />

<!-- <cfsetting showdebugoutput="no" /> -->
```

Listing 2:

```
<div id="voteForm">
<form action="hotfixes.cfc?method=tallyVote" method="post"
onsubmit="this.onsubmit = new Function('return false');">
<table>
<cfloop from="1" to="3" index="i">
```

```
<cfoutput>
<tr>
<td><input type="radio" name="ID"
value="#theHotfixes[i].XmlAttributes.id#"></td>
<td align="left">#theHotfixes[i].article.xmltext#</td>
</tr>
</cfoutput>
</cfloop>
<tr>
<td></td>
<td><br /><input type="submit" value="Vote Now!"
onclick="disableSubmit(this,'voteForm','Thank you!');" /></td>
</tr>
</table>
</form>
</div>
```

Listing 3:

```
<script type="text/javascript">
function disableSubmit(theButton,inputForm,newText) {
theButton.value=newText;
theButton.disabled = true;
var theForm = eval("document." + inputForm);
theForm.submit();
}
</script>
```


tips & tricks

Listing 4:

```
<cffunction name="tallyVote" hint="Registers a vote for a particular
hotfix" output="true" access="remote">
  <cfargument name="ID" hint="The hotfix ID which to add this vote
to" type="numeric" required="yes" />
  <!-- BH: Return the 204 to the browser -->
  <cfset return204() />
  <cfset Request.hotfixes.appxml.hotfixes.hotfix[ID].votes.Xmltext =
Request.hotfixes.appxml.hotfixes.hotfix[ID].votes.Xmltext + 1 />
</cffunction>

<cffunction name="return204" hint="Returns an HTTP StatusCode of 204
to a requesting client. Debugging MUST be turned-off for this to
work consistently." returntype="void" output="true">
  <cfheader statusCode="204" statustext="No Response" />
  <cfflush />
</cffunction>
```

Listing 5:

```
<table>
  <cfset theHotfixes = XMLSearch(Request.hotfixes, "/appxml/hotfix-
es/hotfix")>
  <cfloop from="1" to="#arrayLen(theHotfixes)#" index="i">
    <cfoutput>
      <tr>
        <td>
          <h4>Article #theHotfixes[i].article.xmltext#</h4>
          <a
href="#theHotfixes[i].articleURL.xmltext#"#theHotfixes[i].title.xml-
text#</a>
          <br /><br />
          #theHotfixes[i].desc.xmltext#
          <br /><br />
          <cfif NOT theHotfixes[i].downloaded.xmltext>
            <div id="downloadForm#1#">
              <form
```

```
action="#theHotfixes.cfc?method=downloadHotfix&ID=#theHotfixes[i].XmlAttri-
butes.id#"
method="post" onsubmit="this.onsubmit = new Function('return
false');">
  <input type="submit" value="Download To Server"
onclick="disableSubmit(this,'downloadForm','Downloading...');">
    </form>
  </div>
  <cfelse>
    <a href="#theHotfixes[i].filename.xmltext#">Download to
Browser</a>
  </cfif>
  <hr />
</td>
</tr>
</cfoutput>
</cfloop>
</table>
```

Listing 6:

```
<cffunction name="threadDownloads" hint="Threads all of the downloads
so that they are all downloaded at the same time" returntype="void"
access="remote" output="true">
  <cfset var theHotfixes = XMLSearch(Request.hotfixes, "/appxml/hot-
fixes/hotfix")>
  <cfset var i = '' />
  <cfset return204() />
  <cfloop from="1" to="#arrayLen(theHotfixes)#" index="i">
    <cfhttp
url="http://#CGI.HTTP_HOST#:#CGI.SERVER_PORT#/hotfixes/hotfixes.cfc?me-
thod=downloadHotfix&ID=#theHotfixes[i].XmlAttributes.id#" timeout="5"
resolveurl="no" />
  </cfloop>
</cffunction>
```

Download the Code...
Go to www.coldfusionjournal.com

FREE*CD! (\$198.00 VALUE!



Secrets of the ColdFusion Masters

Every *CFDJ* Article on One CD!

— The Complete Works —

CD is edited by *CFDJ* Editor-in-Chief Robert Diamond and organized into 23 chapters containing more than 450 exclusive *CFDJ* articles!

All in an easy-to-navigate HTML format! **BONUS: Full source code included!**

ORDER AT WWW.SYS-CON.COM/FREECD

*PLUS \$9.95 SHIPPING AND PROCESSING (U.S. ONLY)

©COPYRIGHT 2004 SYS-CON MEDIA. WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE. ALL BRAND AND PRODUCT NAMES ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

**SYS-CON
MEDIA**

Only from the World's Leading i-Technology Publisher

Using OLAP with ColdFusion

Take data analysis to the next level using MS Analysis Services

As a ColdFusion developer, you are no doubt frequently asked for Web-based reports on data such as sales, site hits, project metrics, and other important business information. Users often want features such as the ability to filter or sort by any column, to drill down into the data, or to re-arrange the report to their liking. These can be time-consuming tasks for developers to implement.

This is where Online Analytical Processing (OLAP) comes in. OLAP is a technology for accomplishing all of the above tasks with ease. It is a data engine designed to analyze multidimensional data sets such as sales totals over product family, geographical region, and time period. If you've ever created a pivot table in Excel, then you already understand the basic model of OLAP, as pivot tables are a basic client-side OLAP tool.

To implement OLAP you might think you have to buy expensive client and server tools such as Cognos, Hyperion, or Microstrategy. But if you own SQL Server 2000 you already have a great OLAP setup, because it is bundled and licensed with the companion product, Analysis Services, which I'll shortcut to "MSOLAP."

In this article I'll first give an overview of data warehousing and then briefly show you how to set up the MSOLAP server



By Mark Murphy

and how to work with a data cube. Finally, I will show you how to create Excel spreadsheets and Web pages with interactive OLAP capabilities and how to query a MSOLAP cube via <CFQUERY>.

Data Warehousing and OLAP

Before you dive into OLAP technology, you must first understand the basic concepts of data warehousing. The terms "data warehouse" and "data mart" refer to read-only databases whose purpose is to pull in data from other databases, scrub and normalize the data, and then optimize it for query performance. These functions are commonly referred to as "ETL," for extraction, transformation, and loading.

If you have an environment where you have to pull in information from multiple databases and/or run some data cleansing algorithms, you would be well served to set up an intermediary database. Using DTS packages or third-party ETL tools you can pull and clean the data from multiple sources. Then you point the MSOLAP server to this intermediary database for its data source. OLAP itself is not meant for data cleansing tasks; you should treat data cleansing and data analysis as two different steps in the OLAP development process.

Data warehouses reduce the query load on your primary online transaction processing (OLTP) databases, which should be optimized for the quickest application performance. Data warehouses and OLAP are meant to offload the intensive reporting or data mining-based queries from the OLTP system. OLAP queries usually run very quickly, as many aggregations (a.k.a. "group by" results) are precomputed and stored, so

querying 8 billion rows of source data can be as fast as querying 8 rows, once the data is pulled into the cube (more on cubes shortly).

OLAP systems should not be expected to store real-time data, since refreshing the OLAP system can take time and place a load on the source system(s). You should schedule data refreshes on an interval during nonworking hours.

Getting Started

MSOLAP is installed by running a separate installer from that of the SQL Server; it is located in the /MSOLAP folder of the installation CD. Run the setup program and make sure all the components are installed. Once you've installed it, make sure to apply the latest Analysis Services Service Pack (as of this writing, it's SP3) from the Microsoft site on your server and any administrator clients. Note that this is not the same SP3 as SQL Server; it is a separate package.

Once you have installed and patched the MSOLAP server you can open the Analysis Manager MMC console to work with the server, as shown in Figure 1. With MSOLAP 2000 this is a separate administrative tool from SQL Server Enterprise Manager, but in the forthcoming product version 2005, they will be combined into one tool.

In the Analysis Manager MMC you will see a tree of objects, starting with the server name. Under the server (in this case QBIC1) you will see the preinstalled FoodMart 2000 database. This database is used in the online documentation and in many companion books. I will use the FoodMart 2000 database to explain query options with ColdFusion, and you can then apply the concepts to your own cube.

In MSOLAP, databases are containers for data sources, cubes, security roles, and other shared elements. You can perform administrative tasks such as backups or data refreshes at the database level, so it's best to group each major subject area into a database.

At the next level down in Analysis Manager tree you will see the list of cubes. Cubes are the heart of OLAP, as they are the

containers for the data, relationships, and calculations of your data model. To see the cube elements, right-click on a cube and select "edit" to view the cube editor window, as shown in Figure 2. From this view you can edit the cube or browse the cube's data by clicking the data tab at the bottom.

The tables on the right represent the physical tables from which the data are drawn. These tables are arranged in a "star schema" format, in which a central "fact" table is accompanied by one or more "dimension" tables. Note that this is only a visual representation of the source tables and their relationships; it is not a database view, and it cannot be queried directly. Fields from these tables are used to create cube dimensions and measures, listed in the cube object tree on the left, which are what we query from an MSOLAP data source.

Facts tables contain the numeric fields we want to perform mathematical operations on (such as sum, count, or average). These fields are known as measures in OLAP vocabulary. In our sales example each line item in an order comprises one row in the fact table; each row in the fact table contains the measures store sales, store cost, and unit sales. We can also easily create calculated members such as profit (which is simply [store sales - store cost]) to complement fact tables.

Dimension tables contain the data elements you will use to separate the measures into categories such as product family or geographical region. Dimensions are used for anything you want to filter by or drill down by. You can choose a simple dimension employing just one level, such as gender, or you can have a more complicated dimension such as product, which employs six levels of drill-down detail. Here, note that the gender dimension came from the single customer dimension table only, but the product dimension came from the combination of the product and product class dimension tables. Combinations of dimension tables are known as "snowflake" schemas.

To review, fact tables contain the numerical data (measures) we want returned by MSOLAP, whereas dimension tables contain the categories by which the measures can be separated. At this point you understand the basics of the OLAP data model and are ready to see it in action. Next I'll discuss how to query the server with MS Excel.

Querying MSOLAP with Excel

One of the benefits of OLAP is the ability to allow your users to define their own reports and arrange the data the way they like. Most proficient computer users are familiar with Excel, which is a perfect client tool for querying MSOLAP. As a developer, you need only set up a workbook with interactive pivot table functionality, and the client can use the workbook as a fully customizable query tool. Follow these steps to setup an MSOLAP client workbook:

1. In a new Excel workbook, click Data>Import External Data>New Database Query.
2. On the OLAP Cubes tab click <New Data Source>. Its name can be FoodMart Sales, and its provider is "Microsoft OLE DB for Olap Services 8.0."
3. Click connect, click Analysis Server, and enter the server name. The database is FoodMart 2000, and the cube is Sales.

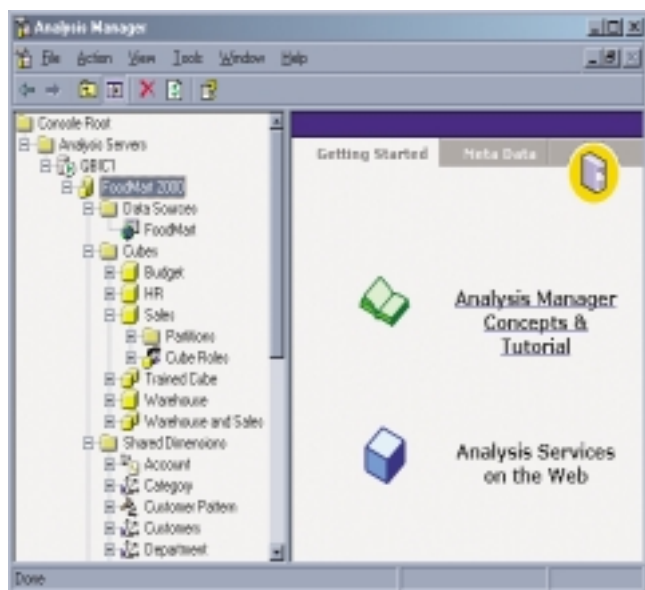


Figure 1: The Analysis Manager MMC console

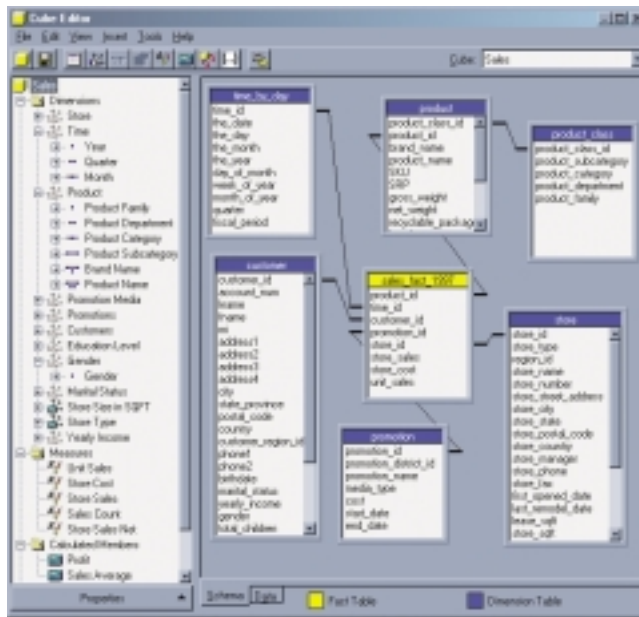


Figure 2: The Cube Editor window

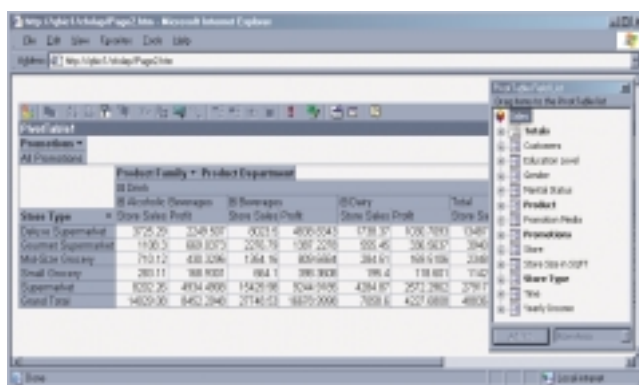


Figure 3: Excel on the Web

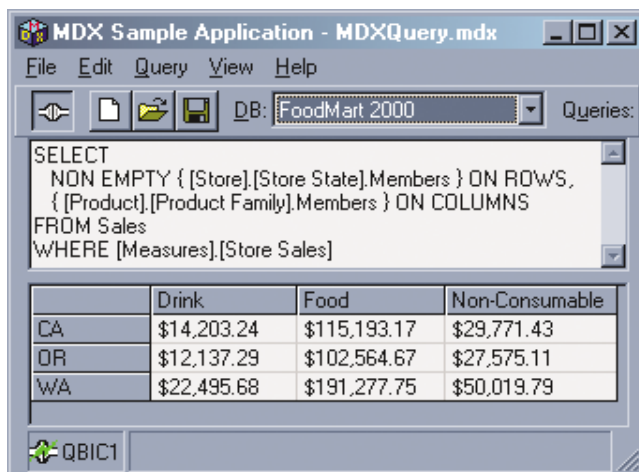


Figure 4: Simple MDX query

- Now you can connect to the OLAP datasource and create a new pivot table in a new worksheet.

You now have a full-fledged OLAP client that your users will like, because it supports drill-down, drag-and-drop pivoting, and filtering on both grids and charts. Users can effectively create their own reports, customize their appearance, and use OLAP data as a basis for other worksheets or reports, with data being automatically refreshed from the server.

Make sure to test whether your users can access the spreadsheet and pivot the data from their machine. They need MDAC 2.7 and the Pivot Table Services drivers. If they are using Windows 2000 or Office 2000, they won't necessarily have the "Microsoft OLE DB for OLAP Services 8.0" driver mentioned in step 1 above. You can use the older driver set included in Office 2000, but it doesn't support all the newest cube features. The easiest solution to get your clients' machines up to date is to distribute the newer driver package (ptsfull.exe) found on the SQL Server installation CD or on the Microsoft Web site.

Excel Over the Web

You can publish those same interactive pivot tables and charts to clients as ActiveX objects embedded in any Web page provided the clients have met the Excel requirements mentioned above and have IE 5.01 or higher installed (see Figure 3). The simplest way is:

- With our pivot table spreadsheet in Excel, choose File>Save As Web Page...>Selection: Sheet, with Add interactivity turned on. Click Publish. Select which type of page to create, and select HTML. Click the "add interactivity" button, and select PivotTable functionality.
- Save this page to your Web server, and give it any Web extension, e.g., .htm or .cfm. The first time it is viewed by users' IE browsers it will show them a security warning. Users can avoid this in the future if they add the site to their list of trusted sites in IE's security options.

The data source connection that you configured in the spreadsheet is hard-coded into the generated OBJECT code, so you don't have to set up the same data source on each client that was set up in the original spreadsheet.

If you want more control over the data and formatting options, you can interact with the ActiveX object using VBScript. The documentation on this is available on the MSDN site; just search for "PivotTable List Control."

Connecting with ColdFusion

Security or client configuration issues may rule out Excel as an option. If so, you can simply use MSOLAP as a normal database to query with ColdFusion and CFQUERY.

The quickest way to connect ColdFusion to MSOLAP is not to set up a new data source, but to reuse your existing data source for SQL Server. You simply set up a linked server on the SQL Server that points to the Analysis Server and passes through the query requests. Then you connect ColdFusion to your normal SQL Server using CFQUERY.

To set up a linked server on SQL Server, just click on Security>Linked Servers>New Linked Server. The driver type is



web services **EDGE**
conference & expo

Web Services Edge
2005 East

International Web Services Conference & Expo

Hynes Convention Center, Boston, MA
February 15-17, 2005

The Largest
i-Technology
Event of
the Year!



**Guaranteed
Minimum
Attendance
3,000
Delegates**

Tuesday, 2/15:
Conference & Expo

Wednesday, 2/16:
Conference & Expo

Thursday, 2/17:
Conference & Expo



Join us in delivering the latest, freshest, and most proven Web services solutions... at the Fifth Annual Web Services Edge 2005 East - International Conference & Expo as we bring together IT professionals, developers, policy makers, industry leaders and academics to share information and exchange ideas on technology trends and best practices in secure Web services and related topics including:

- Transitioning Successfully to SOA
- Federated Web Services
- ebXML
- Orchestration
- Discovery
- The Business Case for SOA
- Interop & Standards
- Web Services Management
- Messaging Buses and SOA
- SOBAs (Service-Oriented Business Apps)
- Enterprise Service Buses
- Delivering ROI with SOA
- Java Web Services
- XML Web Services
- Security
- Professional Open Source
- Systems Integration
- Sarbanes-Oxley
- Grid Computing
- Business Process Management
- Web Services Choreography

3-Day Conference & Education Program features:

- Daily keynotes from companies building successful and secure Web services
- Daily keynote panels from each technology track
- Over 60 sessions and seminars to choose from
- Daily training programs that will cover Web Service Security, J2EE, and ASP.NET
- FREE full-day tutorials on .NET, J2EE, MX, and WebSphere
- Opening night reception

Interested in Exhibiting, Sponsoring or Partnering?

Becoming a Web Services Edge Exhibitor, Sponsor or Partner offers you a unique opportunity to present your organization's message to a targeted audience of Web services professionals. Make your plans now to reach the most qualified software developers, engineers, system architects, analysts, consultants, group leaders, and C-level management responsible for Web services, initiatives, deployment, development and management at the region's best-known IT business address - The Hynes Convention Center in Boston.

For exhibit and sponsorship information please contact Jim Hanchrow at 201.802.3066, or e-mail at jimh@sys-con.com.

Sponsored by:



WebServices



SDTimes

asp.netPRO



NET JOURNAL



WebSphere



All brand and product names mentioned above are trade names, service marks or trademarks of their respective companies.

Contact for Conference Information: Jim Hanchrow, 201-802-3066, jimh@sys-con.com



www.sys-con.com/edge

	[PRODUCT].[ALL- PRODUCTS].[CONSUME]	[PRODUCT].[ALL- PRODUCTS].[FOOD]	[PRODUCT].[ALL- PRODUCTS].[NON- CONSUMABLE]	[STORE].[STORE COUNTRY].[MEMBER_CAPTION]	[STORE].[STORE STATE].[MEMBER_CAPTION]
1	14203.240000000000	131192.16999999999	29775.429999999996	USA	CA
2	11110.209999999997	102544.46999999997	21575.109999999991	USA	OR
3	22495.600000000000	191127.75000000013	50813.709999999998	USA	WA

Figure 5: CFQUERY result

Figure 6: Advanced MDX query

again “Microsoft OLE DB Provider for OLAP Services 8.0,” the data source is the OLAP Server name, and the catalog is the OLAP database name surrounded by square brackets. Once you have this (or any other) linked server configured, you can query that data source via SQL Server using the OPENQUERY function, which passes the query through to the host server. I will demonstrate this capability shortly.

MSOLAP supports a subset of the SQL query language, but you will run into its functional limits very quickly. You are better off using the MSOLAP engine’s built-in query language, Multi-Dimensional Expressions (MDX). It is similar to SQL, in the sense that it has SELECT, FROM, and WHERE clauses, but the query format is very different.

To get started using MDX, you can use the provided MDX sample application shown in Figure 4. In this query we asked for the store states as rows, and the product family as columns. We are asking for a multidimensional recordset, in which rows are not numbered 1-*n*, but actually have data attached to them. Notice that the WHERE clause doesn’t filter the recordset the way SQL does; it is used to specify which measure we want to look at, in this case, store sales.

Listing 1 shows the code to run the same query run through CFMX, and Figure 5 shows the result. We don’t have the option of getting a multidimensional result returned through a standard query, so it gets flattened by SQL Server into a compatible recordset.

In Figure 5, notice that the “rows” data are flattened into columns. In fact, the entire hierarchy of the dimension requested as rows is returned as separate columns, and the text “member_caption” is appended to all the column labels. Additionally, the columns are returned with long, detailed names that are hard to work with. One solution to this problem is to alias the column names in your outer SQL Server query SELECT statement so they’re more useable. But if you don’t know the column names in advance, you won’t be able to


do this, so you would have to add logic to your ColdFusion template to parse the dynamic column names into usable text.

As you can see, additional work must go into setting up the OLAP engine and query capabilities through ColdFusion. For a simple query of store sales by region and product family, the OLAP solution is overkill. But OLAP offers powerful statistical or analytic functions for business intelligence – such as time-series analysis – that are not readily available in SQL.

The next example highlights some more advanced features of MSOLAP (see Figure 6). This query returns the store sales, three-quarter moving average, and difference from the previous quarter, grouped by product category for 1997 sales data. This would be a much more difficult query to develop in standard SQL, don’t you think?

Wrap-Up

At the time of this writing, Microsoft has released its next major SQL Server version (2005) to beta. Reporting and business intelligence are more of a priority for this package, so we will see better ETL tools, tighter integration with SQL Server, and more powerful cube and MDX features. Take a look at www.microsoft.com/sql for further information.

I hope this has been a useful overview of MS Analysis Services and query options over the Web. For more information about getting started, I recommend the book, *SQL Server 2000 Analysis Services, Step by Step*, by Reed Jacobson, published by Microsoft press. You can also find good information in the books online documentation provided with the SQL Server package. You’ll see that creating an OLAP cube can be quick and easy, and can provide you with more data analysis options for your applications. 

About the Author

Mark Murphy is the president of Electric Labs Inc. (www.electriclabs.com), a software consulting company located in New York City. He has been developing ColdFusion applications and data warehouses for over six years and is happy to answer e-mailed questions.

msolap@electriclabs.com

Listing 1

```
<cfquery datasource="QBIC1SQL" name="MSOLAP_Query">
  SELECT = FROM OPENQUERY( FOODMART_2000, '
    SELECT
      NON EMPTY ( [Store].[Store State].Members ) ON COLUMNS
    FROM Sales
    WHERE [Measures].[Store Sales]
  ')
</cfquery>
<cfdump var="#MSOLAP_QUERY#">
```

Download the Code...
Go to www.coldfusionjournal.com

MAX

MAX ATTENDEES
Extend your education with
a **FREE 3 TIME TRIAL**
of ColdFusion

www.sys-con.com/mx/specialsubscription.cfm

Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



- Exclusive feature articles
- Latest *CFDJ* product reviews
- Interviews with the hottest names in ColdFusion
- Code examples you can use in your applications
- *CFDJ* tips and techniques

That's a savings of \$29.89 off the annual newsstand rate. Visit our site at www.sys-con.com/coldfusion or call 1-800-303-5282 and subscribe today!

COLD FUSION Developer's
Journal



ColdFusion

For more information go to...

U.S.

Alabama
Huntsville
Huntsville, AL CFUG
www.nacfcug.com

Alaska
Anchorage
Alaska Macromedia User Group
www.akmmug.org

Arizona
Phoenix
www.azcfug.org

Arizona
Tucson
www.tucsoncfug.org

California
San Francisco
Bay Area CFUG
www.bacfcug.net

California
Riverside
Inland Empire CFUG
www.sccfcug.org

California
EL Segundo
Los Angeles CFUG
www.sccfcug.org

California
Irvine
Orange County CFUG
www.sccfcug.org

California
Davis
Sacramento, CA CFUG
www.saccfcug.org

California
San Jose (temporary)
Silicon Valley CFUG
www.siliconvalleycfug.com

California
San Diego
San Diego, CA CFUG
www.sdccfcug.org/

California
Long Beach
Southern California CFUG
www.sccfcug.org

Colorado
Denver
Denver CFUG
www.denvercfug.org/

Delaware
Kennett Square
Wilmington CFUG
www.bvfcug.org/

Delaware
Laurel
Delmarva CFUG
www.delmarva-cfug.org

Florida
Jacksonville
Jacksonville, FL CFUG
www.jaxfusion.org/

Florida
Winter Springs
Gainesville, FL CFUG
www.gisfusion.com/

Florida
Plantation
South Florida CFUG
www.cfug-sfl.org

Florida
Tallahassee
Tallahassee, FL CFUG
www.tcfug.com/

Florida
Palm Harbor
Tampa, FL CFUG
www.tbmmug.org

Georgia
Atlanta
Atlanta, GA CFUG
www.acfcug.org

Illinois
East Central
East Central Illinois CFUG
www.ecicfcug.org/

Indiana
Avon
Indianapolis, IN CFUG
www.hoosierfusion.com

Indiana
Mishawaka
Northern Indiana CFUG
www.ninmug.org

Iowa
Johnston
Des Moines, IA CFUG
www.hungrycow.com/cfug/

Kentucky
Louisville
Louisville, KY CFUG
www.kymug.com/

Louisiana
Lafayette
Lafayette, LA MMUG
www.cflib.org/acadiana/

Maryland
Lexington Park
California, MD CFUG
<http://www.smdcfug.org>

Maryland
Rockville
Maryland CFUG
www.cfug-md.org

Massachusetts
Quincy
Boston, MA CFUG
www.bostoncfug.com

Michigan
East Lansing
Mid Michigan CFUG
www.coldfusion.org/pages/index.cfm

Minnesota
Brooklyn Park
Twin Cities CFUG
www.colderfusion.com

Missouri
Overland Park
Kansas City, MO CFUG
www.kcfusion.org

Missouri
O'Fallon
St. Louis, MO CFUG
www.stlmmug.com/

New Jersey
Princeton
Central New Jersey CFUG
<http://www.cjcfug.us/>

Nevada
Las Vegas
Las Vegas CFUG
www.sncfcug.com/

New York
Albany
Albany, NY CFUG
www.anycfug.org

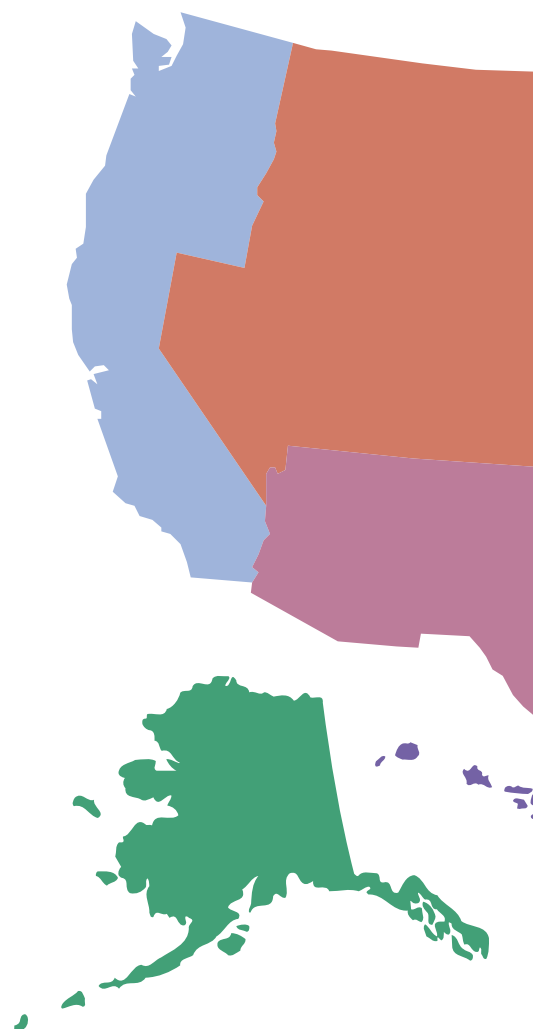
New York
Brooklyn
New York, NY CFUG
www.nycfcug.org

New York
Syracuse
Syracuse, NY CFUG
www.cfugcny.org

North Carolina
Raleigh
Raleigh, NC CFUG
www.ccfug.org

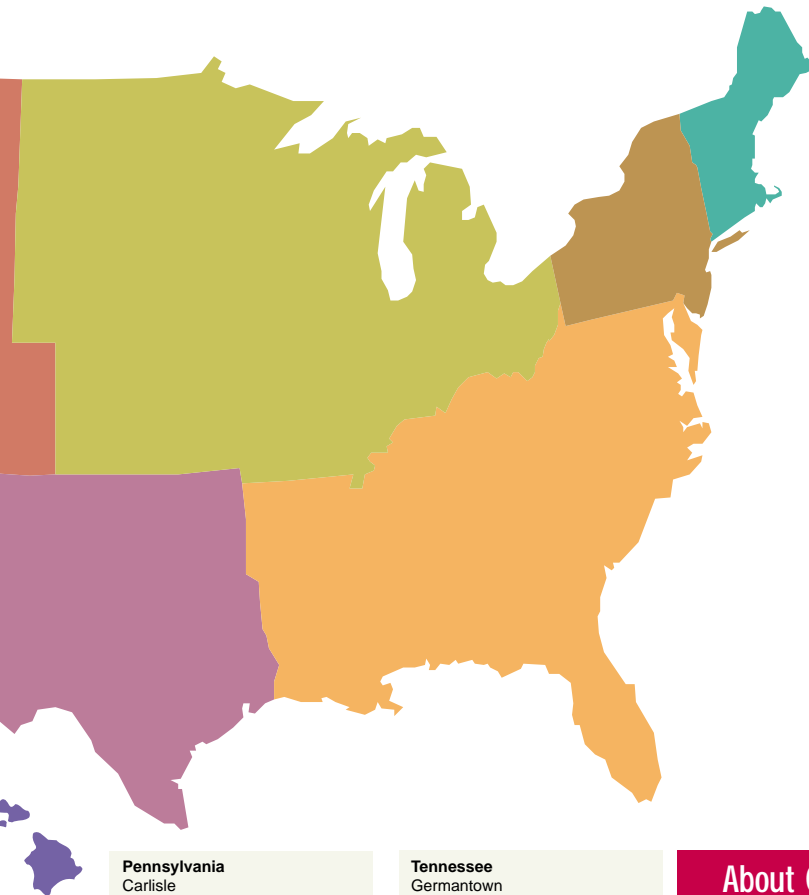
Ohio
Dayton
Greater Dayton CFUG
www.cfd Dayton.com

Oregon
Portland
Portland, OR CFUG
www.pdxcfug.org



User Groups

<http://www.macromedia.com/cfusion/usergroups>



Pennsylvania
Carlisle
Central Penn CFUG
www.centralpenncfug.org

Pennsylvania
Exton
Philadelphia, PA CFUG
www.phillycfug.org/

Pennsylvania
State College
State College, PA CFUG
www.mmug-sc.org/

Rhode Island
Providence
Providence, RI CFUG
www.ricfug.com/www/meetings.cfm

Tennessee
LaVergne
Nashville, TN CFUG
www.ncfug.com

Tennessee
Germantown
Memphis, TN CFUG
www.mmug.mind-over-data.com

Texas
Austin
Austin, TX CFUG
www.cftexas.net/

Texas
Corinth
Dallas, TX CFUG
www.dfwcfug.org/

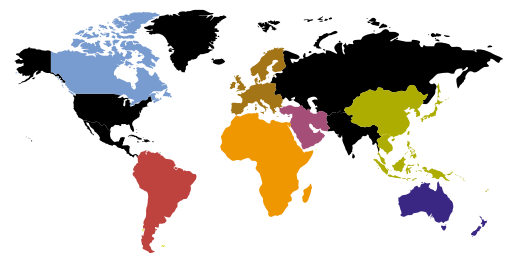
Texas
Houston
Houston Area CFUG
www.houcfug.org

Utah
North Salt Lake
Salt Lake City, UT CFUG
www.slcfug.org

About CFUGs

ColdFusion User Groups provide a forum of support and technology to Web professionals of all levels and professions. Whether you're a designer, seasoned developer, or just starting out – ColdFusion User Groups strengthen community, increase networking, unveil the latest technology innovations, and reveal the techniques that turn novices into experts, and experts into gurus.

INTERNATIONAL



Australia
ACT CFUG
www.actcfug.com

Australia
Queensland CFUG
www.qld.cfug.org.au/

Australia
Southern Australia CFUG
www.cfug.org.au/

Australia
Victoria CFUG
www.cfcentral.com.au

Australia
Western Australia CFUG
www.cfugwa.com/

Italy
Italy CFUG
www.cfmentor.com

Japan
Japan CFUG
cfusion.itfrontier.co.jp/jcfug/jcfug.cfm

Scotland
Scottish CFUG
www.scottishcfug.com

South Africa
Joe-Burg, South Africa CFUG
www.mmug.co.za

South Africa
Cape Town, South Africa CFUG
www.mmug.co.za

Brazil
Brasilia CFUG
www.cfugdf.com.br

Brazil
Rio de Janeiro CFUG
www.cfugrio.com.br/

Brazil
Sao Paulo CFUG
www.cfugsp.com.br

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Ireland
Dublin, Ireland CFUG
www.mmug-dublin.com/

Spain
Spanish CFUG
www.cfugspain.org

Switzerland
Swiss CFUG
www.swisscfug.org

Thailand
Bangkok, Thailand CFUG
thaicfug.tei.or.th/

Turkey
Turkey CFUG
www.cfr.net

United Kingdom
UK CFUG
www.ukcfug.org



Extreme Programming

A (Fr)Agile methodology

The Agile Manifesto is the product of 17 smart, well-meaning developers who met in February 2001 to discuss problems in software development. The list of developers included Kent Beck, Alistair Cockburn, Martin Fowler, Ron Jeffries, Robert “Uncle Bob” Martin, and Dave Thomas – people who have all made substantial contributions to software development.

During the three days the group met, they spoke of the frustration they had with writing software and found a great deal of common ground. From their discussions, the Agile Manifesto emerged in its current form.

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.”

“Agile programming” is the name given to any methodology that subscribes to the Agile Manifesto. The most well-known example of agile methodologies is the set of 12 principles stated by Kent Beck in his book, *Extreme Programming Explained: Embrace Change*. In this article, I’ll explain what Extreme Programming (XP) is and why I think it is so profoundly wrong-headed.

Beck states that XP is founded on four “values”: communication, simplicity, feedback, and courage. Beck says that these values are worked into the following XP practices:

- Test-driven development requires that tests be written before the software is written.
- The Planning Game is used to derive “user stories” – extremely brief reminders to the developers to have more in-depth conversations with the customer.
- Whole team (previously “on-site customer”) requires a continuously available customer to answer questions. This is necessary since XP “requirements” are so vague as to be almost nonexistent.



By Hal Helms

- Small releases assure that every 1–3 weeks a new version of the developing software is released.
- System metaphor provides a way of thinking about the application that takes the place of a documented system architecture.
- Simple design is guided by the aphorism: “Do the simplest thing that could work.” Developers are highly discouraged from actual system architecture, which the extremists pejoratively term “Big Design Up Front.”
- Emergent design is the hope that an overall system architecture will slowly coalesce, based on individual design decisions and an understanding of the system.
- Refactoring is the process of rewriting code as more requirements are introduced and the overall design “emerges.”
- Collective ownership prohibits individuals from owning code. Instead, developers are encouraged to make changes to (refactor) code written by others in an attempt to ensure that the best software is written.
- Pair programming requires that all production code be written by a pair of programmers working on one computer.
- Continuous integration eliminates the problems of late integration by integrating code several times a day.
- Sustainable pace forbids developers from late-night pizza and Mountain Dew sessions, insisting that developers work no more than a 40-hour week.
- Coding standards, the last of the XP practices, make all the other practices easier to implement.

Why is it called “extreme” programming? According to Beck, his inspiration was the idea of “turning up the dial” on readily acknowledged best practices in software engineering. Here’s a sample from Beck:

“If code reviews are good, we’ll review code all the time (pair programming). If testing is good, we’ll test all the time (unit testing), even the customers (functional testing). If design is good, we’ll make it part of everyone’s daily business (refactoring). If architecture is important, everybody will work defining and refining the architecture all the time (metaphor).”

On one of the XP wikis, one impassioned fan put it like this: “XP is the Jonathan Livingston Seagull of programming!” He’s not alone: XP is now a mini-industry, spawning dozens of XP books and a core of fervent followers.

But does XP have it right? If something is good, does “turning the knob up” to extreme levels really make it better? It’s an

Subscribe Today!

— INCLUDES —
FREE
DIGITAL EDITION!
(WITH PAID SUBSCRIPTION)
GET YOUR ACCESS CODE
INSTANTLY!



The major infosecurity issues of the day...
*identity theft, cyber-terrorism, encryption,
perimeter defense, and more* come to the
forefront in **ISSJ** the storage and security
magazine targeted at IT professionals,
managers, and decision makers

SAVE 50% OFF!

(REGULAR NEWSSTAND PRICE)

Only \$39⁹⁹ ONE YEAR
12 ISSUES

www.ISSJournal.com
or 1-888-303-5282



The World's Leading IT Technology Publisher

A new tool for MX professional
developers and designers...

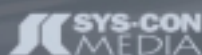


ADVERTISE

Contact: Robyn Forma
robyn@sys-con.com
(201) 802-3022
for details on rates
and programs

SUBSCRIBE

[www.sys-con.com/
mx/subscription.cfm](http://www.sys-con.com/mx/subscription.cfm)
1 (888) 303-5282



MX
developer's journal

important question, for this is the guiding principle behind all of XP's values, practices, and activities.

A logic professor of mine in college used to say that a helpful activity in judging the validity of an assertion is to apply it to a different field. If the principle is really universal, the conclusions should be similarly acceptable. And here, XP fails fatally. Let's try this "wisdom" in some other areas.

"If sleep is good, we should sleep all the time."

"If talking is good, we should talk all the time."

"If working is good, we should work all the time."

These are absurd conclusions: clearly, Beck's guiding principle is not universal. But it may still be that in this particular field of endeavor – writing software – a happy confluence has occurred and that this particular set of practices just works well together.

To determine this, argumentation is futile. If the practices work, the results should attest to them by a success rate that is above the norm. Unfortunately, in reading the impassioned writings of XPers, it becomes clear that the advocates feel that the "higher truth" of XP should be self-evident and does not need to demonstrate actual project success. This may explain why there are so few actual case studies of XP projects. For guidance, we must use the "poster child" of XP projects, the Chrysler Comprehensive Compensation (C3) project, guided by Kent Beck, Ron Jeffries, and Martin Fowler. Certainly, if anyone knows XP, these are the people.

The C3 project was an attempt to take an existing mainframe system and port it to PCs. Chrysler was concerned about the Y2K bug and felt they needed a replacement in case the bug turned out to be real. The project began auspiciously enough in 1995, judging by some of the contemporaneous quotes of the XP advocates who worked on it:

"The best software development team on the face of the earth!" – Chet Hendrikson

"I'd never have said it, but it turns out payroll is so complicated that it is very interesting finding the inner simplicities that make it possible to

write a program that works. Objects are perfect for it. Smalltalk is perfect for it. That's just part of why we call C3 Payroll the best project in the world." – Ron Jeffries

Magazine articles with the XPers took up the refrain: XP had proven itself. Alistair Cockburn, talking with Ron Jeffries in February of 2000, put it like this: "As we talked, I asked Jeffries how success on the C3 project translated into XP use on other Chrysler IT projects. His grin told me all I needed to know."

Well, maybe not quite all of what needed to be known. In February of 2000, the XP faithful were stunned by this announcement on the XP wiki:

"As of February, 2000, the C3 project has been terminated without a successful launch of the next phase."

The "next phase" was something beyond a simple pilot program, the most that the C3 project was ever able to achieve. The much-vaunted C3 project, the best project in the world, was a failure. Again, comments from the XP wiki:

"...at Daimler Chrysler these days the terms, 'C3', 'Extreme Programming', and especially 'the Planning Game'...are now unutterable by anyone wishing management to take them seriously..."

"...in the view of DC's [Daimler Chrysler] management, C3 was a disastrous project and never the like shall be seen again there."

Given the devastating failure of XP in the C3 project, have the XPers moderated their claims? Not at all. The problem, some of the XPers involved in the project stated, was the customer and not the methodology at all. Undaunted by failure, the XP industry continues to churn out books, articles, and unsupported claims at a prodigious rate.

Unable to use statistical analysis – or even numerous anecdotal successes – to bolster their claims, XPers have lapsed into New Age talk.

"Unacknowledged fear is the

source of all project failures." – Kent Beck and Martin Fowler


"Why do we need a software process? For the same reason that we need government, taxes, and laws: fear." – Kent Beck and Martin Fowler

"I had an epiphany. I went back where Kent was and said that we were just 'balancing hopes and fears.' We had focused on our hope that we could launch the system as planned and our fear that we wouldn't. Kent told me that I had just 'snatched the pebble from the master's hand.' We knew what had to be done..." – Chet Hendrikson

Opposing XP today will likely get you branded as an obstructionist, someone who just doesn't "get it." Shorn of actual project success to point to, the XPers have "turned up the dial" on the rhetoric. Alan Cooper, the creator of Visual Basic and author of such books as *The Inmates are Running the Asylum*, had a discussion recently with Beck about XP. **Beck:** For all its Dilbertesque qualities, I'm still proud of having "Embrace Change" in the title of the first XP book. I've consciously decided to give up the ability to predict the future.

Cooper: I see that in XP. It's an abdication.

Beck: Is Zen an abdication?

No, Kent, Zen is a philosophy of life. Wrapping a failed methodology in high-sounding terms doesn't make it profound; it simply obscures the issue. I believe we must demand more of project methodologies than cultish phrases and mystical patter. The ability to repeatedly produce successful software projects is a thing of extreme value – to both programmers and the clients they produce software for. Is it too much to ask that a methodology that promises this actually works? 

About the Author

Hal Helms (www.halhelms.com) is a Team Macromedia member who provides both on-site and remote training in ColdFusion, Java, and Fusebox. Hal is cofounder of the Mach-II project.

hal.helms@teamallaire.com

One damn good Cold Fusion hosting firm!

Webcore Technologies specializes in ColdFusion, ASP and SQL hosting. We offer these solutions in both dedicated server and shared virtual environments. Webcore can design and implement clustered or load-balanced solutions, managed firewalls, VPN's, multi-site failover, and more. In addition, we also offer corporate Internet access, web site design, and technology consulting services.

Our in-house tier 1 level data center enables us to provide the most reliable hosting in the industry. Our Microsoft and Cisco certified team ensures that all support issues are resolved promptly and professionally. Unlike other hosting companies that answer with a phone attendant, you will receive a live person when you call Webcore. No phone attendant, no frustrations, just world class customer service and support the first time you call.



Webcore Technologies, Inc.
877.WCT.HOST
www.webcoretech.com

be the pilot!

FREE SETUP on Shared
Hosting Accounts With
ColdFusion MX Support
Use Promo Code CFDJ2004



INTERMEDIA.NET

WE DARE YOU TO TAKE A FREE TEST FLIGHT!

Managing technology that runs your business is a matter of trust and control. INTERMEDIA.NET gives you both.

TRUST. Since 1995 we have been providing outstanding hosting service and technology to our clients. Don't take our word for it... take theirs.

"The support and service that you offer are nothing short of golden. The high quality of your system and service for CF customers is something one could only ever dream of." – Claude Raiola, Director, AustralianAccommodation.com Pty. Ltd.

CONTROL. We give you instant control over your site, server and account configuration changes. No more submitting requests and waiting for someone else to take action. You are in control to pilot your business through its daily needs.

BE THE PILOT. Take a free test flight and see what our HostPilot™ Control Panel offers you beyond all others. Check out our SLA guarantees. To see more testimonials and to find out about our competitive advantages, visit our Web site at www.Intermedia.NET.

Managed Hosting • Shared Hosting • Microsoft Exchange Hosting

Call us at: 1.800.379.7729 • Visit us at: WWW.INTERMEDIA.NET

