Emerging Technology in Web Development

Daniel Romero

May 02, 2002

Table of Contents

Intro

The standard tools to create web sites have grown exponentially over the past 10 years. At first, it seemed like regular HTML would've been enough, after all, it included tags where the user could display any information they wanted in a way they wanted it. Bold, italic, displaying pictures; were all neat capabilities back then. When the first web page was viewed in November of 1990 it was a certainly a big deal. It was only a few years afterwards however where people had enough of static pages and craved for a better way to exchange information through the Internet. Not only that, but also the idea arose of selling goods and better user interaction and experience. HTML could only do so much, so in the mid 1990s, out came the birth of the server side languages and enhanced multimedia capabilities.

Popular Server Side Languages:

- **Perl** (1987) Originally made for scripting purposes on Unix machines by Larry Wall
- **Active Server Pages** – December 1996 by Microsoft Corporation
- **ColdFusion** – 1995 by Allaire
- **PHP** – 1994 by Rasmus Lerdorf

Popular Breakthroughs in Graphics/Web Multimedia

- **Flash** (1996) – Macromedia (originally by Jonathan Gay, called "FutureSplash Animator")
- **Video Streaming** – 1995 (Microsoft and RealNetworks)

Since most server side languages are similar in concepts and strive for the same goals, we will analyze only one server side language technology: ColdFusion, along with the Flash technology. Later we will see what the future holds for these two goliaths in the web development community.

The Beginning

ColdFusion 1.0 and the later version 1.5 were very simple programs. Their primary feature was database connectivity, through a primitive tag-based script called "Database Markup Language" (DBML). Creating version 1.5 was a fairly dramatic improvement, however; it introduced system service architecture and e-mail integration, and allowed compatibility with C++ for coding extensions. But since it was the first in a new field, ColdFusion soon faced stiff competition from Microsoft."Microsoft ASP was created by a team of developers who were Allaire competitors acquired by Microsoft," Berrey notes. "They had a competing product called DBWeb that was largely a failure in the marketplace, but they were working on a new technology in 1996, called i-Basic, which eventually became ASP." (Quote by Adam Berrey, one of the leading founders of Allaire - http://www.tophosts.com/articles/?3016.html)

ColdFusion has gone through many stages. Currently, it's in version 5, but has certainly progressed since then. It started as a very primitive language, capable of limited tasks. When the second version was released, it supported e-mail server interaction along with extra commands to accomplish a wider task. Later versions started supporting Linux and Unix, and the ability to generate JavaScript and Java on the fly.

How it Works

The main goal of ColdFusion is to link web pages to services that are hosted on a server. That means that it can retrieve information from a database and write to it. This is the key to making web applications. Web applications could not exist if it wasn't for these features.

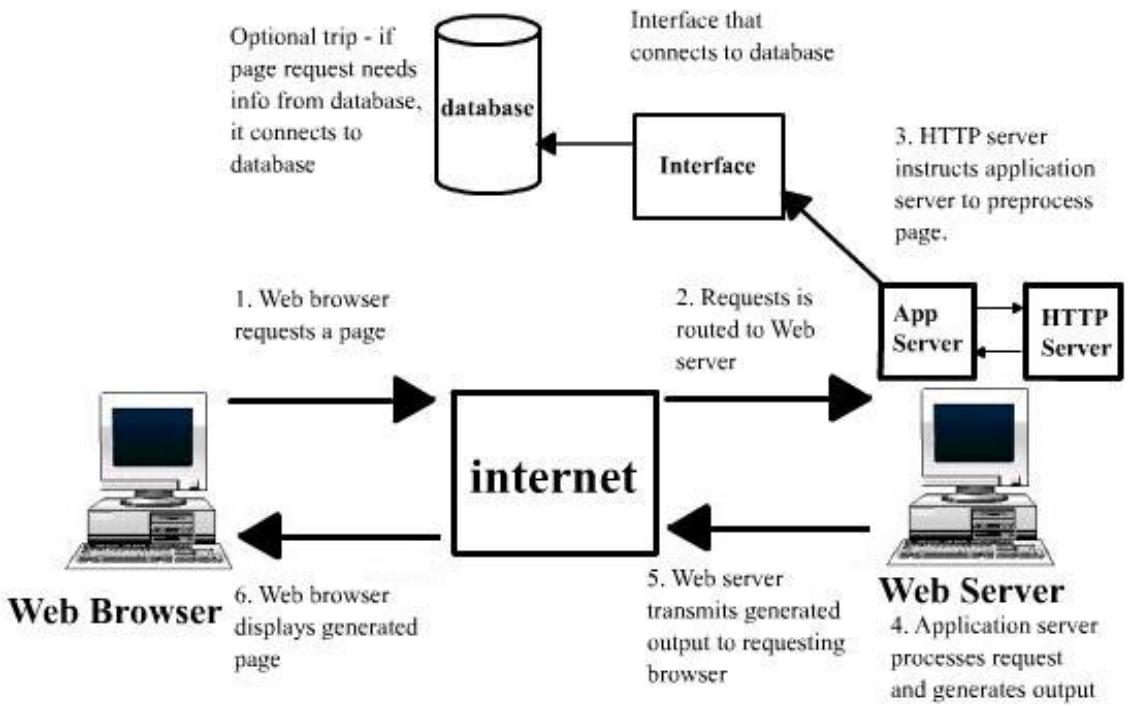
ColdFusion is a web application server that works side by side with an HTTP server. The two are not to be confused. An HTTP server mainly hosts HTML pages alone. They cannot connect to databases or do any complicated processes. This is where a web application server comes in. It stands next to the HTTP server and handles database connections then sends the information back to the HTTP server in order to be sent back to the client and be displayed on the browser. It follows a standard cycle from the client's computer to the server and back whenever there is a page request.

e.g. You need to retrieve your e-mail from your online account

- You put the URL of the web site.
- You make a request to the HTTP server for the page, and connect.
- The HTTP server sees if you need services from the web application server(your e-mail message).
- If so, you connect to the web application server, and then go through an interface like ODBC* that connects you to a database server like SQL Server or a file like Access (your e-mail is most likely stored in a database server).
- You run the query that was provided from the web application server on the database, and go back and work with the HTTP server to create an HTML page with the data.
- You send that data back to the client.
- You get your e-mail on an HTML page.

*ODBC – Open Database Connectivity – is an interface on Windows operating systems that opens communication between a program and database. This is just one type of interface. There are others such as JDBC that connect Java to Java-compatible database services.

The following is a modified diagram from the Macromedia ColdFusion Web Application Construction Kit by Ben Forta and Nate Weiss (et. al). It shows the cycle beginning at the browser, to web server and extending to database (if need be) and back to the browser across the Internet.



ColdFusion uses templates that are written in CFML to connect to databases. These templates control the data that gets sent to the database server and controls what HTML gets sent back to the browser. Don't worry, it will be explained in a moment.

Practicality

How can this database connectivity be put to use? E-mail, as we saw, is one form. Database integration with ColdFusion can lead to a lot of possibilities. The most common web applications out there are the following:

- E-Mail
- Address Books
- E-Commerce
- Tutorials and Academic
- Games
- Chats
- Programs that control huge storage facilities
    - US Post Office Database
    - Government Data
- Organization Recordkeeping
- Content Management

As you can see, all of these applications deal with the manipulation of data as well as storage.

Why ColdFusion?

There's huge list of languages that do (almost) what ColdFusion does. ASP, JSP, PHP, Perl, Python, Zope, MIVA, etc. The list goes on. However, ColdFusion is unique in several ways.

Rapid Development with CFML

To the developer, this will be the most obvious feature. ColdFusion could possibly be the fastest language to develop applications in. ColdFusion supports CFML (ColdFusion Markup Language) which is used to write templates (pages that have a ".cfm" extension that contain CFML code). Within these templates you can mix HTML code with CFML. The neat thing about it is that you can use CFML to control your HTML. You just mix in CFML code (which are special tags that look like HTML that start with "CF") with HTML, and save that as a .CFM file and voila. You have a ColdFusion template. For example, let's say you wanted to write your name in bold 1,000 times on a web page.

You could do:

```
<B>Your Name</B>
<B>Your Name</B>
   .
   .                1,000 times
   .
<B>Your Name</B>
<B>Your Name</B>
```

That would be absurd though. An easier way would be if done you use CFML code:

```
<CFLOOP INDEX="i" FROM="1" TO="1000">
      <B>Your Name</B>
</CFLOOP>
```

So your template will look like the above diagram. When the person goes to access your .CFM template as a web page through the internet, (e.g. http://www.somesite.com/some_coldfusion_page.cfm) ColdFusion gets the signal that there's a page with CFML code being requested, it processes the CFML into HTML, sends it to the HTTP server and it goes back across the internet to your browser. So basically it coverts the CFML code to HTML, instead of you doing the work, it does it for you. The CFML code in this example is looping from one to a 1,000 displaying the HTML <B>Your Name</B>. Remember that you can mix ColdFusion and HTML together.

Why would we want to do this? While this is example might not be realistic to anything anyone might do, it does prove a very interesting point. That you can dynamically make HTML on the fly. And not just make HTML, make LOTS of HTML. With this concept in mind, we can mix data retrieved from a database and automate it with CFML code to produce pages without us designing the web page the traditional way. (Hand coding the web page from scratch using plain HTML.) No ordinary human being would be up to the challenge of coding an HTML page with 5,000 records on it manually. It would be a lot of HTML to hand code. While ColdFusion doesn't decrease the size of the final web page, it does make it faster to actually create the page. Just retrieve 5,000 records from the database with one SQL command, then set up a loop from one to 5,000 and between the loop put your HTML to format it the same way we did in the previous example, and you should be good to go.

So how do you develop faster? Simple, the code isn't as complex and lengthy as others (ASP, PHP). For example, to connect to a database and display the field "user_namex" from every record from table resonance, you would have to do the following in each language:

(This is just for the purpose of comparing code. Understanding the code is not necessary.)

ASP

```
Dim objConn, objRS
Set objConn = Server.CreateObject ("ADODB.Connection")
objConn.Open "resonance"
Set objRS = objConn.Execute ("select * from resonance")

While Not objRS.eof
	Response.Write(objRS("user_namex"))
	objRS.Movenext
Wend

objRS.close
objConn.Close
```

PHP

```
@ $db = mysql_pconnect("localhost", "username", "password");
mysql_select_db("resonance");
$query =  " select * from resonance";
$result = mysql_query($query);
$num_results = mysql_num_rows($result);

for ($i=0; $i<$num_results; $i++)
{
	$row = mysql_fetch_array($result);
	echo $row["user_namex "];
}
```

ColdFusion

```
<CFQUERY NAME="RecordSet" DATASOURCE="Resonance">
	select * from resonance
</CFQUERY>

<CFOUTPUT QUERY="RecordSet">
	#user_namex#
</CFOUTPUT>
```

This example leads to the conclusion that ColdFusion can lower the development time. The CFML (resembling HTML) looks cleaner and less lines have to be used to process data. (Notice that you use the <CFQUERY> tag to connect to a database, and <CFOUTPUT> tag to output data.)

Good Learning Curve

As you saw from the above CFML code, in comparison to other languages, it looks cleaner, and has to potential for less lines of code. It is also easy to maintain in the long run since the code does the not require complicated syntax. It is easier to read.

Build-In Power

ColdFusion is a powerful tool that comes with virtually every feature any programmer will ever need to use. It contains functionality that sometimes on other web application servers you need to install as a separate service. For example, in ASP, if you want to write a small application let the user upload a file, you have to install a component written in C++ to enhance ASP so that it can upload files. This comes build in ColdFusion with the <CFFILE> tag. Other neat features of ColdFusion is its abilities to connect to an LDAP server, send E-mail through SMTP, generate XML on the fly, connect to FTPs, use Java objects, and convert a record set to XML. Few other web application services have all these features built in.

Another unique feature that ColdFusion has is the ability to generate JavaScript and Java on the fly. For example, if you want to do JavaScript form validation on a form, you don't need to know a drop of JavaScript. Just some lines of CFML can do the job. You can also include Java applets in your template. All you do is include the proper CFML code, include some parameters, and the web application server converts the CFML to Java code and puts it on the clients web page.

Probably one of the coolest features is the integration with Macromedia's Flash. ColdFusion, previously owned by Allaire, is now owned by Macromedia (Allaire has merged with Macromedia and no longer exists). This has given Macromedia the chance to integrate the two very nicely.

The Future of ColdFusion Applications

Last year Macromedia announced the news that ColdFusion 6 was coming. (Code named "Neo.") At last year's DevCon (Macromedia Development Conference), Macromedia talked about Neo's new architecture. They told all developers that it will be completely rebuilt and optimized. The web application server processing will no longer run on Macromedia's proprietary engine originally made in C++. It will now use the Java architecture originally developed and still being researched by Sun.

What does this mean? It means speed! Pages will now be processed even faster than the current version of ColdFusion. You will still be able to code using CFML, but your page will now be compiled to java byte code and run on a server that's based on Java architecture. This is what gives it the speed.

Java is fast for server side data processing, especially because of its compiled byte code. When a program is run, it is run by the Java Virtual Machine and uses the Java Runtime Environment. Technology such as this has been researched at Sun for more than a decade; so it has been enhanced to the max. Now ColdFusion will have that power.

In ColdFusion 5 or below, each time a template is requested by the client, it must be processed every time it is requested. It acts as if it was being compiled (your source code to code a computer can understand) every time it is requested. In the new upcoming version of "Neo" this will no longer be the case. When you first create your template, and it is requested once, it gets compiled to java code, and that's the end. You get the compiled code and no longer have to go through the process of compiling. This is a huge enhancement in performance. (If you change your CFML source code, then save it, the server detects a change, and it will recompile again when it is first requested by some arbitrary user.)

The fun doesn't stop there. You will also be able to write templates in Java. This is optional. This however means that you're now allowed to enter the world of the Java community through your ColdFusion templates. You will be able to use the endless amounts of Java classes, packages, beans, J2EE, etc.

The Growth of Flash

Flash has come a long way. Originally intended for simple presentation purposes, it has now grown into a massive multimedia creation package. Flash is well liked because it generates vector animation, which can reduce file size in animations. It does not use bitmap graphics (individual pixels) but rather uses vector graphics, which is complex mathematical instructions to create shapes. For example, if you create an 800x600 image of a square with a black outline and inside filled blue, the bitmap image will have information about each and individual pixel. This can take up a lot of unnecessary space, even if the pixel is the same color. What Flash does is it just stores information about the square, how big it is, and what color it is, and that's it. Of course you need a plugin to view Flash animations, but all it costs is a simple free download.

Just recently Macromedia has started encouraging its developers to use Flash as a web interface to database applications instead of the standard HTML. Last year at the DevCon conference Kevin Lynch, the CTO of Macromedia, asked the public how many used Flash as GUIs to their database applications (made in ColdFusion). It turned out a very low percentage. Not surprised, he told the public his views on the subject. The future of Flash, how "we're trying to go beyond the skip/intros." Then there was a session on integrating Flash with database applications. Macromedia announced it was a good idea to make Flash the graphical interface to database applications instead of HTML for one main reason:

- Compatibility – This is a major issue to take into account when developing a web site. It is virtually impossible to accommodate every user. There are too many browsers out there, all interpreting HTML in their own quirky way. Not only that, some users turn off JavaScript and cookies, which increases the strain on many developers on how to handle the design and security. One must also consider the screen resolution, version of the browser, and operating system.

Dave Watts (A leading ColdFusion developer in the ColdFusion Community) had an interesting statement. He said that when a developer makes a web site, he's pretty much taking a snap shot of time; using tools such as HTML and JavaScript that change through time, along with the browsers that interpret them. When the developer makes a web site, he's basically writing a web site specifically for that

frame in time. Not knowing what will happen with the changes in browsers, or any specific technology he used for his web site.

In Flash, this problem is solved. From the start Flash has tried to stay compatible to avoid the browser problem when viewing HTML. No matter what OS you use (as long as you have the plugin), no matter what browser you use, no matter what resolution you have it at (Flash can expand keeping the same ratio), Flash will appear the same. This provides developers with a powerful tool: a true WYSIWYG development environment. Not WYSIWYHYG.

Almost Perfect

It will be a while until Flash replaces HTML for developing graphical interfaces. Although tempting, there are some complications. The main problem that exists in Flash is the learning curve. It's easier to learn HTML and integrate it with other languages than it is with Flash. Also, simple objects like textboxes, check box, radio buttons, that are easily created in HTML, have to practically be created from scratch. There is more work in creating an interface in Flash than in HTML. One must also be accustomed to using Flash's timeline concept and be familiar with its own language called ActionScript.
Macromedia just released the new version of Flash (Flash MX) recently. A new feature they added in this version is (wouldn't you know it?) form objects. Now you can drag and drop your own text boxes, radio buttons, check boxes, buttons and selection boxes all on your movie. This will save a lot of time when creating applications. This however does not decrease the development time. The developer still has to know flash and ActionScript to accomplish anything good. Aside from this feature, they have extended their compatibility with XML. Now Flash can communicate with other applications through XML.

Flash is still relatively new to be used as a graphical interface for database applications. Flash MX is certainly a stepping stone however.

The future for these two look promising. There is a large community for ColdFusion and Flash, and I'm certain they'll grow.

Opposition and Turn Offs

Price

One thing stands out when using ColdFusion – the price. Most application servers out there are free. ColdFusion is one of the few products that come with a price. The professional version of ColdFusion is about $1,500 and the enterprise version is about $5,000. This could be a turn off to some small businesses that are starting off. Why pay for ColdFusion when something like ASP comes free when you install Windows server?

Learning Curve

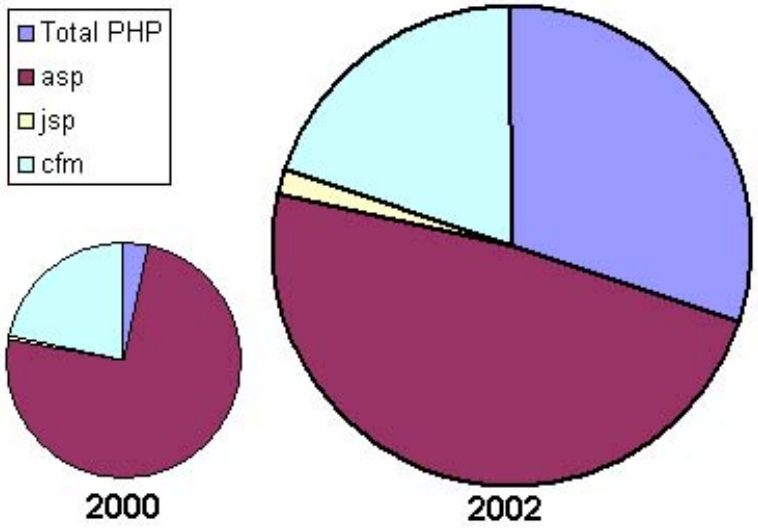
While most developers agree that CFML is easy to pick up, it can be a turn off to others. ASP, being one of the most popular languages out there (beats ColdFusion, PHP, and almost all the others), still maintains to be the tool for developers. Why?

Most R&D departments in companies have Visual Basic programmers, if not, use a lot of Microsoft related tools. That means that the learning gap between Visual Basic and ASP is very slim. If they don't use Visual Basic, they use C++, which is similar to JSP, PHP, and more related to Perl than it is with ColdFusion.

Community

There's no doubt that the community for ColdFusion is big. There are many sites that extend the languages and functionality of it. There are conferences all over the U.S., magazines, books, etc. The list goes on. However, this community may seem rather small compared to giants such as: Java, Perl, and ASP.

In a research study by Dan Bodenheimer, the popularity is as follows:



PHP	157,470	7,549,230	4694%	30%
ASP	3,166,710	11,958,185	278%	48%
JSP	24,435	413,827	1594%	2%
CFM	936,223	4,950,133	429%	20%

Just a note: PHP = ("Personal Home Pages" HyperText Preprocessor) ASP = Active Server Pages, JSP = Java Server Pages, CFM = ColdFusion.

Source: http://php.weblogs.com/popularity

Companies Utilizing Coldfusion

ColdFusion although not perfect, still stands strong in the industry world. The following companies use it for the reasons previously stated: rapid development and easy database integration. Here are some popular companies that integrate ColdFusion in their business:

- FAO Shwartz
- 7UP
- LEGO
- Crayola
- HotBot
- Timex
- Nike Mindsprint
- Hard Rock Hotel and Casino
- K-Swiss
- Xerox
- Starbucks
- Kodak
- Macys

List is gathered from
http://dynamic.macromedia.com/bin/MM/showcase/scripts/showcase_cs_listing_by_query.jsp?product=ColdFusion

Final Thought

ColdFusion remains to be a competitor in the web development world because it's easy to use, and simple to integrate with databases. You can start building applications in no time compared to the other goliaths. The outlook on this product looks very good because soon it will enter the Java world; something many developers are eager about. It will increase the speed dramatically and will enhance its array of features. In few years, I predict, ColdFusion, along with Flash will make an even killer combination.

References / Work Cited

Caillau, Robert. A Little History of the World Wide Web. W3C. 1995.
http://www.w3.org/History.html

Forta, Ben, et al. The Macromedia ColdFusion 5 Web Application Construction Kit. Indiana, US: Que, 2001.

Gay Jonathan. The History of Flash. Untold History. March 4, 2002.
http://untoldhistory.weblogs.com/stories/storyReader$4

Horton, Ivor. Beginning Java 2. Birmingham, UK: Wrox Press, 1999.

Schwartz, Randal L.. Learning Perl. California, US: O'Reilly, 1997.

Wille, Christoph. Sams Teach Yourself Active Server Pages in 24 Hours. Indiana, US: Sams Publishing, 1999.

What are Active Server Pages (ASP). Programmer's Resource.
http://www.programmersresource.com/articles/whatisasp.asp