# Coldfusion IDEs Part II - Projects Vs. Work Orders

Posted At : January 18, 2008 1:41 PM | Posted By : Mark Kruger
Related Categories: Coldfusion's Future

Many of you enjoyed (or were irritated by) my **previous post** on the "Great Coldfusion IDE Debate". In that post I introduced Warren - a Coldfusion developer who approaches development from a technical and practical viewpoint. Many of you pointed out that Warren was a stereotype. Thanks for noticing. I would add that water is wet, Bill Gates is rich and Dick Chaney is a bad shot. Of course it is worth noting (as I did previously) that you need the qualities of *both* a technical and design oriented developer to make a complete skill set. Meanwhile the point of the previous post was that when we talk about a new "Coldfusion IDE" we are not *really* targeting design folks. Instead we are targeting technical coders who are involved in creating CFCs, interacting with databases and data and doing the "heavy lifting" on the server. In short, we are targeting Warren. In this post I would like to talk a little more about some of the challenges a technical coder faces in development, and how such challenges relate to a chosen IDE. In particular I would like to talk about the dichotomy between *high end* and *low end* development - between "projects" and "work orders".

*NOTE: As before, remember that when you comment you do so as a guest. I realize you may have strong feelings on this topic. Just keep in mind that this is* **my** *bully pulpit. Now back to our regularly scheduled post.*

### Challenge Number 1: The New Project

If Warren worked for me he would often be required to work on a project as part of a team. For example, we recently developed a technical charting application for equities. The application included Flex for drawing the charts and studies and implementing behaviors like zooming, scrolling, cross hairs, and customizations. The studies were things like moving averages, MACD, OBV, Bollinger bands, DSS etc. These studies require recursively examining large arrays of data points and running complex algebraic equations. For this task we used Java. Finally, the project required CFCs to handle the interaction of the Flex charting application with the data on the server. The application was then integrated into a subscription toolkit. So to accomplish a project like this required a Coldfusion programmer, a designer, a Flex programmer and a Java programmer.

If Warren was put on such a team as the CF programmer he would need to understand the whole system and his part in it. His process would be to use CFE to create CFCs, check code in and out of Subversion and create test frameworks for his CFC code that serve as examples for the Flex programmer. In this example, a "project" approach works great. All of the cool stuff that comes with an IDE for check in/check out, working directories and versioning fits into the plan. In this case the code is all deployed to a dev server so there is no "production" or "staging" server (yet). Basically we can run a build deployment anytime and test.

### Challenge Number 2: The "Version 2" Project

Three months down the road we have our charting application in production and everyone is happy. At this point we have a "Dev" server and "Production" server. The customer comes back to us and says "We want to add intraday charting". What's our next move? One thing would be to create a staging server for load testing and QA. Now

I have 3 servers - dev, staging and production. In some shops there is a fourth tier of "local dev" where each developer is running a copy of the code on their own local workstation. Now we are working with the dev server and as we near deployment of our new fancy pants "intraday" application we will deploy to the staging server for thorough testing. Note, we did thorough testing before, but we did it on the dev server. Because it was "new" project the use of a third server seemed superfluous. After all, our first "production" build was at launch time.

The challenges of a version 2 project are built on top of the challenges of a version 1 project. Not only do we have to think about builds, versions and dependencies but now we need ANT or something similar to manage multiple deployments. We also need to think about regression testing. Again, Warren will find the project stuff in eclipse helpful in this regard. Eclipse, after all, is built (along with the relevant plugins) around the idea of this kind of team development with QA, version control, builds etc. One thing to note, version control is actually as important to this sort of project management as the IDE being used.

**Challenge Number 3: Work Orders**

So you are on a team. You have lots of resources at your disposal and you have a big fancy 2000 hour project that you are collaborating on. Good for you! Now let's talk about money. To be sure there is money to be made on large projects. But in our shop we also make money from what we call "maintenance" customers. These are sites that may or may not be hosted with us. They may or may not have their own servers. In fact, I would wager that only a small fraction of all web sites, ecommerce stores, applications and the like ever go through anything like the process described above. It would be my guess that the majority of web sites (including Coldfusion sites) are built by individuals and *not* by teams. I would wager that most of them are not using source control and do not have a dev server (other than a developer's workstation perhaps). When deployment time comes they use FTP to put files on a shared server at a commodity hosting company. Sometimes these developers have their own way of doing things that may not make top 10 "best practices" list.

Should developers working this way change how they are doing things to conform to the controls and constraints above? In some cases they should. We recently acquired a customer with a huge code base - an ecommerce store bringing in over 3 million a year. There was no dev server, no staging server, and no source control. The site resided on a shared host with many other sites. Such a site is ripe for a new approach based on measured and carefully planned development, QA and source control - which is why they came to us. However, thousands (probably millions) of sites are not as large, not as complex and (here's the real reason) do not generate enough revenue to support an expensive project effort. For the same reason that folks go to Meineke to get a new muffler instead of the dealership, these site owners will continue to use the thousands of independent contractors out there who are comfortable with an unencumbered way of doing things.

Indeed, we have about 20 or 30 customers who fit into the "maintenance work order" category. They call us with 2 or 3 hours worth of changes every few months. In our case we have a dev server where we have the customer "preview" the changes. But we do not have a great number of internal controls. Often, deployment is done through FTP. These customers need certain things done quickly and *cost effectively*. We make money by satisfying this need without overburdening the customer with expensive process and (what he or she will see as) bureaucracy.

## Why not a KISS Editor

For these second tier "maintenance customers" a CF Webtools team member might do 3 or 4 work orders per day. The team leader might review code from a dozen such changes in a day. So a single developer or team leader might access 8 or 10 different servers in a day. What *they* need in an editor is not an integrated project tool. What they need is to be able to *open, edit and save files*. They don't need to set up a "site" (a la Dreamweaver) or a new "working folder" (a la eclipse). They need to be able to browse the file system, make a change, save the file, and test the change. They should be able to do this whether the file is local, on a network drive or remote using FTP or RDS. In short, they need the editor to *get out of the way*. They don't want an IDE that "helps" them by trying to change the way they do things. This is why my biggest complaint about Dreamweaver is actually the awful file browser it foists upon you. Folders and files are in the same long window where you must drill down into the files using the scroll bars. It reminds me of the registry editor.

Of course DW doesn't really *want* us to think about "editing files". It wants us to use the fancy template system, wizards, local and remote sites, design views and property panels. For example, it doesn't want me to edit my CSS file directly. It will put it in the appropriate spot, pat my hand and say "there there", and direct me to use the tiny panels to adjust CSS files (why are all the property panels so small and hard to read?). Well excuse me - sometimes I want to open a file, edit a file and save a file. That's what I want to do. I don't need 5 roadblocks to changing the phone number on the contact us page or adding a column to a report.

On a similar tack Eclipse users often lament the deficiencies in FTP for the same reason. Yes, there is not an FTP plugin (there is more than 1). It's just that I cannot seem to simply "open a file using ftp" and "save it back to the server". I have to have workspaces, local copies and etc.

So I'm going on record as saying, that while I understand that the "integrated" in IDE is important and essential to project efforts, there is also a necessity for a lightweight editor that does not impose any particular methodology on a developer. It should be possible to use the editor to open a file (either local or remote) make a change, and save the file. It should also be possible to easily browse the directory and file list of a local, network or remote drive.

## Acceptance Is the Last Stage of Grieving

Now before all you masters in computer science and project management types add comments excoriating me on my tolerance for bad practice let me add one more thing. The reason I am sanguine about it is that I have accepted the fact that this situation is not likely to change. Jesus said to his disciples "...the poor you have with you always" (Mathew 26) - and if I can paraphrase Him without incurring a targeted weather event, I would say "the poor website you have with you always." The "low end" development effort will *always* be an important part of the web. As long as GoDaddy is selling hosting packages for 9.95 a month someone will be ready to throw in their Grandpa's $1500 inheritance and hang out a shingle on the web. You can kick and scream, but in my view this is a need that should be met and embraced. If it wasn't for those poorly written sites that can't scale, where would our customers come from? Let it go. Don't become bitter over your lesser cousins editing files directly on production servers. Just think of it as future revenue waiting in the wings :)