Taking Over Third Party Code

Posted At : March 8, 2007 11:54 AM | Posted By : Mark Kruger
Related Categories: Project Management

It happens to us all the time - perhaps every few weeks. A new customer comes knocking and they have an existing application that needs help or support. Sometimes the system has simply outgrown the current developer or the skill set of a contractor and it needs a team. Other times the application has evolved over time and has reached it's peak for performance and is beginning to degrade. Sometimes a rift has developed between the site owner and the previous developer or company. Whatever the reason, taking over third party code can be a daunting task. I have a few tips on how to make the transition more smooth.

Don't Be Afraid

In the word's of Gandalf at the gates of Minis Tirith, "Remember, whatever comes through that door... you are men of Gondor!" Of course immediately after he said that trolls burst the door and started throwing them like pillows at a sorority party. I doubt you will face anything quite so intense. Still, it is important to remember that it is just code after all. You *are going to be able to read it and figure it out*. If a customer has settled on you or your team as a solution it is probably because you are equipped to do it.

Start With Security

Especially if you are replacing a former team or a former developer, you will need to be able to provide assurances to the site owner that you are in control of the site. One of the first things to do is to review the usernames and passwords for low level items like datasources, the CF Administrator, Active Directory, services etc. Don't forget to go through the site's "admin section" (wherever they do their work) and review all the accounts to make sure they are what the site owner intends for them to be. I usually start with a checklist that works through all the servers and applications I know about. Be prepared. After making these changes you will find out about other applications that you have "broken" by changing passwords or usernames and you will have to fix these as well. Make sure that the site owner knows that this process is messy.

Dev Environment

Many sites do not have a developement or staging environment with which to work. Sometimes the site has just not had the attention it needed as it grew, or perhaps it has evolved from something small to something much bigger. One of the most important things you can do to increase the level of confidence is to rehost the code to a dev box and use subversion or some other form of source control. Your customer will love being able to test and approve changes without affecting the live server.

Utility Script Search

Every site that has been alive for more than a month has at least one or two of these. Take a look at the directories and open files that have names like "fixcategories.cfm" or "tmpSetUser.cfm". You might find DB routines or file routines that handle some bulk task or another. Don't delete these files since they may be global fixes for problems you will run into later (ha). But you might want to gather them into a single folder and put a "cfabort" at the top of each of them.

Customer Relationship

If the customer (the site owner) has just gone through getting rid of a team or developer in favor of you or your team, they probably had some good reasons to do it. If the previous relationship was strained you might find yourself on the backside of a messy breakup. The site owner may be prone to fears that would never occur to you about code ownership, mis-use of customer data, and general integrity. Your job will initially consist of smoothing out the rough edges of the process and building a relationship. Yes, I said building a relationship. You may have to do some-hand holding (metaphorically speaking of course). You will have to assuage fears, build confidence and "over-communicate" until his or her trust is rebuilt. If the customer *has* gone through a messy breakup with a former developer or team, here is some specific advice.

- **Don't Bad-mouth the Former Developer** - Even if your customer is searching for a hit man to rub him out, don't criticize your predecessor. It is so easy to commiserate with customers who have seen their business suffer because of a developer who didn't live up to billing, or perhaps did something downright malicious. Examine any codebase and you are likely to find things that are ugly and need revision, but do not fall into this trap of blaming and pointing fingers at an easy target. Remember, you have probably written your share of crap too. Pointing out other developer's flaws will ultimately do nothing to build a good working relationship with your customer, but it can make you look petty and unprofessional. Instead, adopt a conciliatory posture. Comment on positive things. Show compassion and empathy. Taking the high road allows you to operate "above the fray" and that is an important attribute - especially if you are an outsource provider.
- **Over-Communicate** - I hate it when developers say "I don't want to get too technical". Being technical is why you are paid well. You are tasked with providing technical answers and doing technical things. I usually end up providing 2 answers to everything - a summary of information that is reasonably understandable by a lay person and a "technical explanation". Yes, it is true that many customers gloss over the "details", but providing it helps build confidence and (more importantly) often provides a trail of documentation on the changes you are making.
- **Realistic Expectations** - The big question that a customer often has for CF Webtools is, "If we sign with you will this solve our problem?". Now I'm not much of a salesman. I hate marketing speak. I'd rather poke my eye out than meet with my own insurance guy. But even I, as marketing challenged as I am, find it hard to resist promising what I cannot deliver. If an application is suffering a new developer or team can be the answer, but that does *not* mean things will get better in a day or even a month. It might be months before we begin to reach goals set initially. We might end up recommending an entire rewrite. What we promise is that we will bring a high level of expertise and compitence to the table and strive to do whatever is best for our customer.
- **Cost Expectations** - Along with the point above, it is important to set realistic expectations as to cost. Follow the axiom, "It is better to over estimate the cost and come in under budget." If you do this you will have a customer who may wince at the beginning, but will be increasingly favorable toward the end - rather than the downward cost spiral of so many projects. Under no circumstances should you ever provide ad-hoc "ballpark" figures. Cost estimates are tied to estimated hours or resources which are tied to actual requirements. You are not

selling carpet cleaning. One of the surest ways of sinking a project (or in some cases your business) is to be fllip about estimates. Do your homework.

Don't Try to Do Too Much At Once

Finally, in regard to taking over third party code it is important to do triage. You are not going to fix everything or bring it up to your coding standard in a weekend (you do have a standard don't you? And please tell me it includes a note about the proper use of pound signs:). It may pain you to hear it, but there may be sections of code that you never change. It's true. Sometimes bad legacy code is more cost effective to leave in place.