Legacy Code Part 17: The Rest of The Story

Posted At : September 24, 2013 5:52 PM | Posted By : Cutter
Related Categories: Legacy Code, Development, ColdFusion, Application Setup, Adobe

As developers, sometimes it is hard to see beyond the code. We know code. We like code. Some of us occasionally even dream in code (just ask my wife...) Knowing this, it is sometimes difficult, when troubleshooting performance of Legacy Code, to remember that the issue *might not* be in code.

One great area, that so many developers overlook, is the ColdFusion server itself. The installer makes it fairly easy to get things up and running, but is the environment truly setup optimally for your application? By default, CF only loads allocating 512kb of RAM to the underlying JVM. 512kb is not much, especially in a high traffic, heavy process system. And have you ever adjusted your Request Tuning settings? Checked your Queue Timeout? What sort of template caching are you using?

Another area to look at is your JVM itself. While you can adjust some of the JVM settings in the CFIDE (in ColdFusion 10) properly, it's still a good idea to review things like your Garbage Collection settings, your RAM allocation, and other bits. Are you loading unnecessary libraries? And, are you still using the installed version of the JVM? It might be worth while to download a new JDK, install, and test your app. For one thing, it'll help keep your underlying Tomcat more secure.

Then there's the hardware and network review. What sort of throughput are you getting from your hard drives? Is your file I/O optimal for the ops you're running? Are you storing data and assets on other network connected systems and, if so, what sort of throughput are you getting from those internal connections.

And then we're back to your database. Hopefully it isn't on the same machine, but the same sort of review applies. Do you have enough RAM? What sort of hard drives are you using? Are you getting enough throughput through that old NIC card?

Some of these things are really easy to address, taking a little time and effort and research. Others may require parts and equipment replacement, maintenance windows and downtime. It's really up to you to balance out what makes sense.

Step one: write a plan.

Step two: follow through.

With these last 17 posts I've written down a lot of little things to help anyone take their Legacy Code (outdated and tired ColdFusion applications) and begin to carry them forward into new life. Some changes are farther reaching than others, and all require

careful research, planning, management and testing. To let these applications sit, without thought for modern change, is to waste the time originally spent to create them in the first place. Nothing grows without care and feeding and light.

It is my belief that you can still benefit from the hundreds of thousands man hours of previous time and effort spent on these applications, without having to take drastic moves that could prove unnecessary or even disastrous. ColdFusion was *the first* web application server, outliving many predecessors, and continues to grow with the times. ColdFusion continues to be a fantastic Rapid Application Development platform, allowing many developers to write applications in a fraction of the time it would take on other platforms, and there's a solid roadmap for future versions already on the board. There are thousands upon thousands of active, thriving, scalable and performant ColdFusion based applications on servers in thousands of top ranking companies in countries all around the world, still, to this day, because of the power available here.

This article is the seventeenth in a series of articles on bringing life back to your legacy ColdFusion applications. Follow along in the ***Legacy Code*** *category.*