Becoming a Better Troubleshooter

Posted At : March 21, 2006 2:52 PM | Posted By : Mark Kruger
Related Categories: Podcasts, Coldfusion Troubleshooting

Every week I seem to find myself dealing with intractable bugs or performance issues for CF Webtools' customers. Last week, for example, I found myself troubleshooting a JVM for a CF 7 customer, a Database performance issue, a JMS issue and a persistent memory leak in a COM object. That's a pretty typical week for me.

I like troubleshooting and debugging. I suppose it's the Sherlock Holmes in me that likes to pour over minute details looking for clues and possibilities. I think a good troubleshooter has that quality in his nature - the thirst for knowledge and the desire for intellectual growth. I would say that's one of my strengths. That is not to say you can't be a good troubleshooter without those skills, but it helps if you really enjoy uncharted territory.

Listen Here

Before I give you my 4 tips for successful troubleshooting I have one overarching tip that comes to me by way of the world of parenting - be patient! Sam Gamgee's gaffer told him "short cuts make long delays". The job of troubleshooting is not like coding. It involves research and investigation. Take your time to find out all the stuff you need to know (and then some). Your solution will be more thoughtful, and in the end, you will know more about the system for the next time you have to solve a problem.

So… with your patience in hand I offer these 4 tips for troubleshooting a sick server or application.

1. See the big picture first

Before you get down to fiddling with the application scope or reinstalling drivers, take the time to examine the whole system. Take a close look at all the components that make up the server and it's applications. So JRUN is at 100% - does that mean that Coldfusion is the culprit? What about the database? What about 3rd party applications? What about the integrity of the code? How about capacity planning? There are a great many things to examine before you jump to any one conclusion. Take the time to see the big picture.

- Where is the problem occurring
- What processes are involved
- Is there a direct line in your reasoning.
- What are *all* the processes used on the sytem?

2. Check the mundane

Like many folks in IT I started out in tech support. My first job was supporting a server application for salvage yards. Salvage folks would connect to the server (running mumps) using terminals and flat modular cable. A fellow called me one day with a monitor that was bad. I checked the notes and this was the second monitor in 2 weeks. He was irate… "why do you keep sending me lousy monitors!"

Naturally, being the sophisticated technician that I was, I suspected a bad serial port on the serial bus. I walked the user back to the server room and had him plug his cable

into an alternate port - no dice. The monitor would still not light up. So I had him switch it with a "known good" monitor to make see if the port was live. Sure enough the known good monitor was working fine. It must be the monitor. I sent him an RMA and replaced the monitor.

Exactly 1 week later he called and asked for me by name - usually not a good sign for a support technician. "This *#$(* monitor you sent me is out again" he informed me heatedly.

"No way" I said. There is no way that 3 monitors at the same location could fail in 3 weeks time. I told him I would call him back. I went to the store room and got a terminal monitor of that exact make and model and brought it to my desk. I set it up on our test system and then stared at it's physical makeup for a few minutes.

Slowly a smile spread across my face as I realized what was going on. I called the user back and asked if the monitor had been moved recently. The customer thought for a moment and then said no, but they had remodeled the counter three weeks ago. I asked him to describe where the monitor was placed on the counter. He said it was on the left side of the counter against a small shelf or wall that had been installed to showcase refurbished parts.

Then I asked him to make sure the monitor switch was on and reach his hand along the underside of the monitor on the left and find the small dial. Turn the dial away from the wall. It was the "bright" switch. Turning it all the way on immediately illuminated the monitor which had been working fine all along.

Each Sunday the cleaning company cleaned the office. Salvage yards are filthy dusty places. Until the wall had been installed dusting the monitor had had no effect - presumably because they were dusting in the "bright" direction. When they installed the wall, however, the cleaning person had to dust in the "dim" direction and he or she had been dimming the monitor once a week. The lesson - check the little stuff. We spent a few hundred dollars in shipping and monitor replacement because nobody thought to check the dim switch. Don't forget little stuff like:

- power issues
- driver issues
- configuration issues
- usernames, passwords

Another important point on the "little stuff" - read the error messages! See if this sounds familiar to you.

Client: Mark, I'm getting an error message every time I click on the link!

Mark: What does it say?

Client: I don't know, I clicked "OK".

Mark: Well click on the link again….

Client: Ok… I did it

Mark: Now what does the error message say…

Client: …. I don't know….

Mark: Why not?

Client: I clicked OK….

The "OK" button is like magic. Clients think clicking "OK" will make everything "OK". When they see error messages they don't think to read them. Sometimes they are even useful error messages.

Client: Mark I can't log in!!

Mark: Ok, what does the system message tell you when you try?

Client: "Your username is not found"

Mark: Ok, that means the username you are using is not in the database.

Client: But I keep trying it over and over again… look, I'll try it again… see, "not found".

Mark: Ok… I'll look up your username. Meanwhile I want you to take your sedatives and make an appointment for your therapist…. Ok?

Client: Ok…. Uh…. Does she know my username?

Error messages and debug information are very useful. Some programmer (who actually wrote the software) has taken the time to give you a heads up on what he or she thinks is wrong. You should listen. You need to know where the log files are, how to turn on the debug information (how does anyone manage to program without the debug on??) and how to read cryptic error messages and decipher the content.

3. If "easy" solutions present themselves test and try them - otherwise, list and prioritize.

One of the biggest problems with troubleshooting is that there is sometimes not a single solution that is readily available. Instead there is a list of *possible solutions* that must be tested one by one to *see* if they solve the problem. Once you have the list there are several questions to ask that will help you deal with it.

- Are any items on the list easy and straightforward to test and implement? If they are, you might try them first - even if you suspect they are not the "real" solution. If the situation is extremely urgent you have to use wisdom, but in general trying an "easy" fix that does no harm is a good first step.
- Do you have a suspicion as to the item that is *most likely* to solve your problem. If so, put it at the top of the list.
- How hard are your solutions to implement. This can affect the order as well. Sometimes I am trying to make the *minimum amount of changes* to a system. In that case, less intrusive fixes are more appealing.

There's no substitute for experience in this area. If you don't have sufficient experience

there are a host of folks out there in cyber space who are willing to help - which leads me to my next point.

4. Use Internet Resources

Ask questions to the dev lists (**CF Talk** is a good place to start, but there are many other sites and forums). Try ask-a-muse at the top left of this blog. I might just have an answer or know someone who does. Use Google. Amazingly, just about everything has been written down by now and it's all available on line. Use the documentation. Amazingly, many vendors have an interest in their application and it's performance.

Follow these tips and you will find that you can effectively troubleshoot even complex problems regardless of temperament. One thing to note, these tips are pretty broad. There are some "troubleshooting checklists" that give you some narrowly defined parameters for dealing with specific kinds of problems. I've found these lists to be useful - although they are largely just common sense. Happy troubleshooting.