March, 2002

Editor's View

Control Amid Chaos

Remember the days when you could totally control what was on the machines that ran your applications? When you didn't have to worry about interactions with other software or hardware?

I had another reminder recently just how gone those days are. Another developer hired me to automate the production of a particularly complex report. The report in question required intricate formatting that was beyond the capabilities of VFP's Report Designer. In fact, even in Word, it was far from easy.

Before beginning, I confirmed with my client (the other developer) that the target for this task was VFP 7 automating Word 2000. I set to work, and many hours later, had the thing working with the sample data I'd been provided. I sent it off to my client, indicating that it needed to be tested with a much larger data set.

That began several weeks of back-and-forth with the client. He couldn't get my code to run consistently. The first time he ran it, all was well, but subsequent runs bombed with an OLE error. I couldn't replicate his error.

As we worked our way through the issue, the first thing I learned was that he was running not Word 2000, but Word XP. Unfortunately, I didn't have a machine available on which I could install Word XP that could also run VFP 7.

After a lot of back and forth, we determined that whatever was going on was a timing problem. When he stepped through the test program, it worked fine, but running it at full speed failed. Sure enough, adding a brief WAIT TIMEOUT just before the command that caused the error solved the problem.

But that was only a patch. We really needed to know whether this was a problem in Word XP, or some other difference, so the other developer would know what to tell his client, the ultimate user of this application.

I asked around among the Word and Office experts I know and Beth Melton, a Word MVP, came up with the answer. The problem wasn't with Word at all, but with the Office Plug-In component of Norton Anti-Virus. It slowed things down enough that the automation process was failing. Sure enough, when my client disabled that plug-in, the problem went away.

This leaves us with a larger problem, though. How do we ensure that the client doesn't run into this problem or a similar one? My client can't control what anti-virus software his client uses. He can certainly tell them that NAV's Office Plug-In will cause problems, but can we be sure that's the only potential problem?

My code isn't doing anything unusual. The line that fails looks something like:

```
oDocument = oWord.Documents.Add( cTemplate )
```

That's pretty standard automation code. If another product that's not even part of this process can make that code fail, how can we ensure that any code is safe? My guess is that code that performs computations within a single product is probably unlikely to run into issues, but any code that talks to the file system or interoperates between multiple applications seems to me to be vulnerable to this kind of problem.

Unfortunately, I don't have any solutions to suggest. I don't know how to protect the code I write from other applications running on the same machine. Let me know if you've found the surefire cure.

VFP 7 Service Pack 1 available

We're already reaping dividends from the decision to separate VFP from the other Visual Studio products. The first, of course, was that VFP 7 shipped last summer, while Visual Studio .NET has yet to ship. By the time you read this, the second tangible bonus should be available-Service Pack 1 for VFP 7.

With VFP 6, which was a part of Visual Studio 6, the first service pack that included anything for VFP (SP3) didn't come out until a year after the product was released. Then, it was another year until the next service pack and more than 6 months after that until the final service pack (SP5).

With VFP on its own, it was about 6 months from the product shipping until the first service pack. Since Microsoft released hot fixes right away for a few serious problems, this seems like just about the right time frame – long enough to let people find the problems, but fast enough to keep them from being a major headache.

So what's in this service pack? In line with Microsoft policy, it's almost entirely bug fixes. There's one minor enhancement-XMLToCursor() has a new flag that lets you add data to an existing cursor.

What's fixed? A variety of bugs, most of them minor, but a few that have been showstoppers for people. Probably the highest profile item is a problem that leads VFP itself or a VFP app at runtime to stay on top of other applications.

A number of fatal errors (including some of those nasty C0000005 bugs) are fixed. I've found VFP 7 to be pretty stable to begin with, so from my perspective, this service pack makes VFP 7 perhaps the most stable version of FoxPro ever.

If you're already using VFP 7, you'll want to download (from msdn.Microsoft.com/vfoxpro) and install the service pack right away. If you're one of those cautious folks who likes to wait around for Microsoft to iron out any problems, your wait is over. Go ahead and upgrade now, then download the service pack. (Note that the service pack is available only as a download; it won't be integrated into the packaged product.) Once you start working with VFP 7, you'll wonder what you were waiting for.