From the Editor

Mission: Possible

Whil Hentzen

As you read this, the calendar says 1997. Orwell's *1984* happened 13 years ago. Hong Kong is going to revert in a few months. And if the movies are at all accurate, we'll be walling off Manhattan and throwing all the criminals into it later this year. A big spaceship named Discovery will be heading toward Jupiter in a couple of years (2001). And Captain James T. Kirk is going to take command of his own starship in about 325 years.

Oh, yeah, one more thing. Virtually every computer in the world is going to crash and burn in about 33 months. (Yes, even those running Access 95.)

But that's not all. A lot of applications out there are on their last legs. We don't have to worry about the gazillions of mainframe apps because, fortunately, experienced COBOL programmers are a dime a dozen and just dying for a chance to maintain some 1967 code. However, those old FoxBASE and Clipper (S'87) apps reliably toiling away in DOS don't work too well in DOS boxes and their users are getting tired of exiting Windows to run them. Others run okay under Windows—they're just ugly. But they still don't play well with the other boys and girls in the GUI world. And let's face it: The business world has changed in 10 years. The pace has quickened—and the business rules hard-coded in those apps are hopelessly out of date. The apps have been hacked and kludged to death, and using them is sometimes more trouble than going back to stone tablet and chisel. Do you want to hear a really scary statistic? There's over $80 billion of live applications for which the source code can't be found. (You thought you were the only goofball who did that, right?) And over the last 12 months, a new delivery mechanism has arisen: The Internet and its associated technologies have been brought to the forefront as a new way to centralize data and applications but, at the same time, to provide inexpensive access to a larger community of users.

So these four factors—Year 2000, character mode and DOS apps becoming outdated, business rules changing faster than the apps can be modified, and the Internet delivery model—represent an incredible opportunity. The opportunity is accentuated because these factors are happening at the same time. Together they present a compelling reason for upgrading and replacing these applications.

Insurmountable opportunities

One of the most famous cartoon lines of all time is from Pogo: *We have met the enemy and he is us.* Many people don't remember the next line: *We are faced with insurmountable opportunities.*

1997 will be the year that forward-thinking MIS managers look at the state of their systems, consider the factors outlined above, and start to (gasp!) plan for the future. We're expecting a number of calls from firms looking to begin conversion of systems.

Why should they start now? Well, some may be of the opinion that planning is a positive attribute. We'll let them live in that dream world, okay? But we need to face a couple of fear factors as well. First, the supply of developers is more plentiful now than it will be in another 18 or 24 months—when *everyone* will need their systems rewritten. And not only will it be harder to find experienced, qualified developers, but they're going to need them even worse—because as the supply and demand curve becomes tilted, profiteering will raise its ugly head. Yes, one side effect of this increased demand will be that these corporate developers will scratch their heads and ask themselves, "Why am I sitting here in this cubicle, making some paltry salary, and getting screamed at and dumped on because my managers didn't plan ahead? Why should I stay here and work till midnight for the next year to handle some emergency that wasn't my fault, when I could be out there as an independent and making a ton of money?" So these corporate folk may walk away, exactly at the time when their companies need them most.

Have I presented a convincing case that demand for our work will go through the roof?

What are we going to do about it? I know I've been wrestling with this situation, and many of you will do the same. The problem is that a lot of people have been rather disconcerted over the past year and a half or two.

Every time a radically new product is introduced (new product or new version—heck, it's all the same thing), development takes a pause. Kind of like when the new model Corvette is introduced—the purchasing

of the old ones slows. Of course the old version doesn't disappear; some people like it better. (The Sting Ray is still the best looking 'Vette of all time, right?) Some people want a deal and don't feel that the square taillights and different knobs on the air conditioner are worth $3,500. And still others just don't have a clue—they aren't aware that Chevrolet introduces a new model each year, so they just buy whatever's on the lot. (They also pay sticker without asking <g>.)

But most people play the Wait And See game, at least for a while. And they drive Ol' Reliable—the '69 Dodge Dart that's held up through three wars and six presidents—until they decide…

Well, we've kind of been waiting for a year or two with [Whil: should this say "for" instead of "with"?] the new version of VFP. Version 3.0 just didn't quite feel right. Sure, it had a lot of really great features. OOP that was out of this world. Control over forms that we could only dream of. The basic rudiments of a data dictionary. True Windows. But there were still a few things that just annoyed me, and made dealing with the product less than the Zen-like experience I was waiting for. The Debugger, for example. Win32s. As long as this kludge was available, customers asked us to use a Ferrari to plow their back 40. We could, but we didn't really want to. And then there's, well, hey—need I go further? Weren't these two reasons enough to stay in version 2.6 for a while?

Insert new subhead here?

Our shop sold four version 5.0 apps in the couple of weeks before DevCon in Phoenix, and we're getting a call a week from more people interested in FoxPro work. We've examined 5.0 pretty carefully, and while there are a couple improvements we'd like to see, we're really happy with it.

We've taken a good look at the competition. And to quote Alicia Silverstone: "*Develop in (fill in the blank)? As if!"* Life is too short!

However, development in a new version is a major commitment of resources. Another reason we declined to spend a lot of resources when version 3.0 was released was the tepid backing of the FoxPro marketing folk. We've seen where VFP 5.0 (and future releases) are being positioned. We can make a more educated guess about what 1998 and 2001 might bring, and feel comfortable that VFP is going to have a slice of that market. We feel confident that David Lazar and Robert Green will bring verve to the previously moribund image. And we see that customers are increasingly comfortable as well.

Don't want to be Pollyanna, to be sure. The various hordes in Redmond will continue to leave VFP off slides and some of their sales folk will say simply the stupidest things. Frankly, if corporate Earth were to wholesale adopt VFP in the next three months, we'd all be in big trouble. How would you like to have to turn down new work every four days? It would just kill you, wouldn't it?

Wait a minute! I just presented a picture where demand is going to shoot through the roof over the next 18 months! What are we going to do? Can we wait until VB 5.0? Delphi 3.0? PowerBuilder 5.0? Oracle99? These tools are just going to keep evolving. It's time to set some roots down and begin to develop apps that (1) will last for a few years and (2) can be extended as new technologies appear.

I think VFP 5.0 is the ideal platform for developing applications for the next three to four years. Here's why:

1. The technical merits of the tool are numerous. It's difficult to find anyone who has anything bad to say about VFP 5.0. Well, I'm sure we could find one loose cannon in the Pacific Northwest, but, hey, he's busy enough trying to buy a product that should have been buried in 1989.
2. Given what we've seen in public about VFP 6.0, as well as the current intentions of Microsoft and others in the industry, the use of VFP as one of the tools (not the only one, however) for the late 1990s is solid and robust.
3. There are no worthy competitors. Microsoft is continuing to pour resources into VFP, enhancing it and integrating it with the other development tools in the Visual Tools suite. ActiveX integration. Internet Server capability. Common tools.
4. Customers are responding considerably more favorably to VFP 5.0 than 3.0.
5. Last but not least, having people on the VFP marketing team who actually care about the product for a change is rather refreshing as well, wouldn't you say?

So what tool are we going to use? One that chokes on record sets of 100,000? One that comes from a company that hasn't made money in two years? One that requires a $3,000 outlay per developer just to click on an icon? One that requires $1,000 per user for the necessary operating system and back-end licenses? In

many cases, Fox is an excellent choice. It’s fast, it’s inexpensive, we know how to use it, and it’s going to grow with us.

The mission is possible.