## Column: From Editor
## Figures: NONE
## File for Companion Disk: None

Patience!

Whil Hentzen

We've got a "really-big-shew" for you tonight, but before we start, let me tell you about what's coming next week. We've got a lot of great stuff coming – XML applications that Fox developers could finally care about, another president at Compaq Computer, palmtops, handhelds, ebooks, settops, mobile communicators, and web-enabled pagers (although none of them talk to each other very well), yet another service pack for Windows NT 4.0, a new chip that holds four and a half bazillion transistors, and more. So be sure to tune in next week as well.

Tonight's big "shew"

An editorial in Computing Canada back in 1997 contained my most favorite quote of all time – even better than Ernie Hudson's "Ray, when someone asks if you are a god, you say 'YES!!!!!'" classic from Ghostbusters.

"IT projects are not engineering projects. Software research is a better description. In engineering, a repeatable process is applied to a problem such as building a bridge. The technology stays the same, the tools stay the same, the process stays the same. If any of these change, the bridge goes over budget. With software projects, the technology changes every project, the tools change every project and the process is in a continuous state of flux. Add to this mix that what is likely being built is the automation of a likely poorly understood process. All that should be known is that with all the unknowns, any estimates are educated guesses."

So, obviously, the one thing we're not going to see on next week's week show is software we can trust.

I bring this quote up because these next few years are going to be tough ones. Customers, once they get past the turn of the century in a few months, are going to be increasingly demanding. They've been reading about enterprise-enabled applications, components, multi-tier systems, smart-everythings – and will figure that it's time they got involved. These aren't cutting edge technologies anymore, they figure, they must be mainstream by now. And so it's time to call up their favorite developer and get rolling with some projects.

And you're going to have to explain that the Emperor is still walking around purt near nekkid. Keep this quote in mind the next time you're involved in a "Death March" (Ed Yourden, Prentice Hall; ISBN: 0130146595), and late one Friday when they back you into a corner of the conference room, demanding, "Well, Mr. Smarty-Pants? Why isn't this system up yet?"

Let's look at this quote in a bit more depth.

The technology changes every project, the tools change every project.

We've actually been rather fortunate – our tool of choice has been fairly stable for the last few years. No mind-wrenching changes in paradigms – the object model, the DBC, and back-end database connectivity have all stayed pretty much the same. Some of you have over four years of solid Visual FoxPro experience under your belt now. Your class libraries are robust, you know every OLE error message, and what to do, by heart, and you spend most of your time in new development, not in maintenance of old stuff that wasn't put together quite properly.

But that's about it for stability. How about Microsoft's data strategy of the week? What version is MDAC up to now? 2.5? 2.6? 2.814a? I think I've seen seven versions in the last 18 months. I'm sure glad I didn't have to ship anything that used ADO 1.5 – it would have driven a customer crazy. "Oh, you'll have to install this new update for the system I installed a few months ago. Why? Well, the data access components had a few bugs…" Imagine repeating this line seven times in a year and a half! Good bye customer!

Web technologies, of course, are much worse – everyone's playing hide and go seek with "What's new this week." The problem is that, unlike the goofballs who did this with automobiles 90 years ago, our people can just ‘flip a switch” and implement a new idea. When your ideas had to be transformed into smoke-belching metal, you had to take longer and think things through.

And everyone in the industry is to blame – it was a VP from Sun from whom I heard the term “perpetual beta.” Everyone is looking over everyone else’s shoulder, wondering if they’ve got the next big thing. And so they rush things into production before they’ve even thought through the design, much less coded or tested it properly. How can you produce reliable systems in a repeatable fashion when the tools you’re using are “yesterday’s news” by the time you’ve finished downloading the file from their website?

So face it – the technology has already changed twice by the time this newsletter reaches you. You can either ignore it (dBASE III+ wasn’t *that* bad, was it?) or decide to accept it. But don’t keep your customers in the dark. Just as it’s your job to educate them about why certain interface designs are bad ideas, it’s also your job to educate them about this situation – let them know they’re riding on a wild horse along with you.

The process is in a continuous state of flux.

Are you doing things the same way you did them five years ago? Not unless you’re an idiot! You’ve gotten smarter, learned better ways, developed techniques that save you time and money, and are more reliable than last week’s tricks.

You’ve had to, because there are no “generally accepted practices” in our business. You can show a balance sheet or a ledger put together by an accountant in London to another accountant in Los Angeles, and it will make sense. You can hardly look at code you wrote six months ago without wondering, “Why did I do it *that* way?”

We’ve made a lot of progress in the days since Codebook introduced the Hungarian naming conventions – anyone remember what your code looked like in FoxBase days? But we’re all still inventing things as we go. Is it any wonder that two ActiveX controls made by the same manufacturer that work beautifully in Visual Basic 6 produce terrifically disparate results in Access or Visual FoxPro? Those manufacturers are just trying to get the things to work.

Again, it’s your responsibility to keep your customers informed. Let your customers know that you’re making it up as you go – *just like everyone else is*. They’re not hiring you for 20 years of perfect jobs – they’re hiring you because you’re committed to continual improvement and you’ve shown you can learn from your past mistakes.

Automation of a poorly understood system.

This one is a bit more delicate, because it’s largely in your customer’s court, and you have to be nice when pointing out problems on their end. You may not be able to ask them why it took them three meetings to determine how they want to move inventory from the receiving dock to a holding warehouse. I mean, they’ve been doing it for 37 years – why don’t they know by now?

It’s OK if they don’t – and you have to make sure they understand that. Just as rotten documentation is our industries’ Achilles Heel, poorly understand processes (or, often, processes that are bypassed in the heat of the moment) are the Achilles Heel of many of our customer’s businesses. Reassure them – the analysis of a process requires them to define the process, and that means they have to make decisions. A lot of people are reticent to do so, for fear of making a mistake that ends up becoming very visible. And some of those decisions aren’t binary, and thus are hard to make for a computer to replicate. When humans ran those processes, they could apply fuzzy logic and make value judgments easily.

Well, that’s all for tonight. I hope you enjoyed the show, and learned something you can use when juggling between customers, software projects, uncertain technology - and lil’ ole you in the middle. It’s going to be a long few years, and I imagine a few of you will figure it’s easier to dig ditches for a living. But with this perspective on the relative maturity of our industry, you’ve got one more weapon when push comes to shove with your customers. Along with a bit of patience, it may be all you need. It’s certainly all you’re going to get.