from the editor

Last Man Standing: Our Dirty Little Secret

Whil Hentzen

If you haven't seen it, I highly recommend the Bruce Willis flick, Last Man Standing. Well, that's if you've the stomach for the merger of The Meanest Men in the West and The Terminator. Not a lot of plot twists, not too much fancy dialogue, no involved character development, but buckets of testosterone and an awful lot of empty shell casings by the end of the first scene.

Watching Bruce empty magazine after magazine into the bad guys coming down the stairs and through the door made me reconsider the need for what many of us take as a given - the need to be able to handle and reconcile conflicts in multi-user applications. Actually, not having a lot of first-hand experience with firearms, I thought about this after I stopped wondering how he was able to pump seventy or eighty shots out of a single revolver. Those cartridges must hold more bullets than they appear able to.

Just exactly why do we have to worry about reconciling conflicts within a record between users? We've spent so much time trying to play nicey-nice - and to what avail? When does this ever come into play? We assumed that we had to do it, just in case the user in Topeka is editing the street address and the user in Belfast is editing the city of the same record.

Every time I've ever heard this topic discussed - either in print or at a lecture - the presenter starts out with "Let's suppose that..." I've never heard "In an app I was writing last week…" Okay, I heard it once, but I knew the speaker was lying - they hadn't actually written an app in years.

Well, I've a modest proposal: just keep overwriting all of the changes until the last user is done. The last user is the one who wins. The Last Man Standing. Seriously, after taking into account some simple record locking techniques to make sure that the user doesn't get one of those annoying "another user is editing your record and you're just plum outa luck until they come back from lunch" messages, I think we may be done.

There are well over ten thousand of you reading this column (well, to be more accurate, there are ten thousand of you who *have the opportunity* to read this column <grin>), and let's be honest - how many of you really run into the situation where one of your users is updating Field A in a record, and another of your users is updating Field B in the very same record, and you have to make sure that both changes are saved at the same time? It's our dirty little secret - we simply don't have this need all that often.

Yes, we know we can do this. We've published articles on the techniques with GetFieldState and DBCGetProp and all sorts of other fancy tricks so that you can detect and handle these changes. That's why we're here - to show you how to do the highly technical stuff (that's a technical word itself, by the way - don't use it indiscriminately.)

But in the immortal words of Andy Griebel, "That's too hard for me to do." (I wish I could duplicate his Arkansas twang and easy-going grin at the same time.)

Or rather, too complex, for a lot of situations. We've written well over 20 systems in the last three years, and they're running in dozens of states and a few foreign counties - and we've not yet once run into the situation where the customer has been concerned about this problem - much less run into the situation where the customer actually wanted us to do something about it!

I suppose we could 'write it once and never look at it again.' Yeah, but if it's not needed, then it's not being run. And if it's not being run, it's not being tested. And so, two or three years from now, it will blow up. And you'll have no idea what that code that you haven't looking at since the last presidential election is doing.

And try explaining why your SAVE() method is 215 lines long to a customer who wants to take over routine maintenance.

I'm not saying that no one would ever need to do this. It's pretty easy to come up with examples where you would need to. And if you're handling transactions, then this could become a given immediately. But it's also pretty easy to spend a long time in the business without ever traveling down that road.

Don't throw those old issues of FoxTalk away - as sure as shooting (had to put that pun in here…), the minute you do, you'll need that article about conflict resolution because a customer is going to require it.

We've had several, including one just a couple of months ago, that gave you the tools when you do need it. But don't feel that you have to do it every single time - because no, you're not the only one who decided to go with "Last Man Standing" conflict resolution.