
One Last Compile...

The Client/Server blues

I don't know what you lot are doing for Christmas, but I shall be gloomily sitting underneath the Christmas tree reading books on SQL. Client/server has raised its ugly head over the top of my cubicle, and suddenly the world seems a less friendly place.

Up until now, y'see, I've always been a Paradox person. And, touch wood, I've never had problems with it. While others wail up and down the length of Cix conferencing about indexes getting out of date and general data mayhem, the most hassle I've ever had is the occasional *multiple net files in use* message. I've yet to lose a single record through the BDE letting me down, and that's despite the fact that sometimes I even forget to set LOCAL SHARE to True. No, my biggest complaint about Paradox has been that it hasn't got a cool name. It sounds like a kiddie's database, and I don't think it looks good on my CV. Even dBase sounds better: more obscure, vaguely Star-Trekky, something that only clever computer people like me can make sense of.

All of which is irrelevant, because somebody has rashly

decided to buy one of our database systems and install it on a Wide Area Network. It runs like treacle. Well, actually, it doesn't run at all, at least not in any remotely usable form, and so the customer has instructed us to do a client/server version. So Paradox is out, and in comes whichever of the backend databases best matches our client's business requirements, ie which works out the cheapest in terms of user licenses.

My boss assured the customer that our architecture was sufficiently flexible for us to do this quite easily. So it is, if you define 'easy' as redesigning all forms which use DBGrids in your application (about twenty), replacing all the TTables with TQueries (about two hundred), then writing a bit of SQL (several thousand lines, give or take). A breeze.

All of this would have been enough to guarantee me a fairly gloomy Christmas on its own, but I suspect my problems aren't going to stop there. This WAN, you see, is based on ISDN. So it's not particularly speedy. Even shifting as much as possible onto the server isn't going to make the thing usable. I've

done a bit of reading about this, and in a book on Client/Server development I found some tips on performance optimising for databases on WANs. The main bit of advice? Brace yourselves: *don't use any data-aware components*. They all take up bandwidth, you see, even the measly little DBText component. So the suggestion is to use standard Edits and Labels, and do the reading and writing from the database yourself.

Yes, yes, I know some of you are saying that you don't use data-aware controls anyway because they're not flexible enough and you never trust a component that you didn't write yourself in assembler because you're that tough, but for me, this is the end of life as I know it.

I'm not going to worry about porting this application any more, because there's nothing to port. I can't reuse any of the forms. I can't reuse the data modules. The splash screen looks like it might survive, but I'm not counting on it.

The only plus is that at the end of this I get to put something like 'familiar with Microsoft SQL Server' or 'Experienced in InterBase Server' on my CV, but at the moment that seems like scant consolation, I can tell you. Merry Christmas? Bah, humbug.