Are you a Delphi developer, or a Windows developer? (http://www.sandon.it/node/86)

Posted on: Thu, 03/13/2014 - 00:27 **By:** ldsandon

This is not a rethorical question. And not because now Delphi targets OSX, iOS or Android, and it applies to those writing Delphi applications that runs on Windows only. There's a real difference, and it's important to understand it especially now the usual lame BorInCodeDero marketing is trying to use the end of Windows XP support to sell Delphi upgrades (http://blogs.embarcadero.com/stephenball/2014/03/11/are-you-sleep-walking-into-xp-end-of-life/#comment-2316). But is Delphi today a real Windows development tool? My answer is no. Today it is a tool covering just a small part of the whole "envelope" of Windows development, the "client desktop application, with some limitations".

The root of this problem lays in the origin of Delphi itself, and its Turbo Pascal ancestor. Back then, the main Microsoft OS was a single-user single-thread one, with limited network connectivity. Windows was a GUI built atop it. x86 servers was mostly running Netware, a few Lan Manager, Windows NT was too new - and Borland products didn't support any of them. Delphi was developed for Windows 3.1, and its target use. But Microsoft had far broader plan for Windows. Even if Windows 95 took the stage, the real revolution was the NT line - which included a real server version, full network support, and a stronger security model. NT4 was the first really "usable" version, but since Windows 2000 Microsoft offered a full range of OS version able to cover most commercial needs, and since 2003 64 bit it could also break into realms before held by *nix OSes. Today Windows no longer means a "GUI environment atop DOS", but its the name of a full range of operating systems, from embedded ones, desktop, servers, hypervisors.

But did Delphi evolved consequently? Nope. BorInCodeDero management has been stubbornly fixed into the "desktop environment" perspective since them, and unable to understand enough what the evolution of Windows should have meant for Delphi also. Some timid attempt to take advantage of the new "server" perspective was made (Datasnap), but the "server world" was always something beyond - they got an RDBMS server with the dBase acquisition - but never understood this kind of server would have killed the "shared file" DBs. They bought Visigenic, probably more because IT fashion than dictated you needed a broker, than to build a sound tool to build server-side object oriented applications. They didn't even build TCP/IP support into the VCL - relying on volunteers efforts to deliver it through a 3rd party library - Indy (aftear a couple of deprecated socket components). And never understood the NT line was introducing a more complex OS, but able to deliver those features (security, manageability) required by fast growing networks. The boastful "Client/Server" and "Enterprise" versions of Delphi, after all, didn't offer more than some wrappers over **client**-side libraries for server applications. All the required server, client and communication layers were implemented by the server application vendor, not Delphi. Delphi was just a front-end development tool.

The failure of Delphi to deliver a good web development solution could be ascribed to this failure to understand server-side development. This is inherently different from end-user client-side applications. It requires to write scalable multi-threaded applications, while most front-ends,

especially in the past, can work as single-threaded applications with very little scalability needs. That means a good memory manage and network library, and a decent integration with the OS/network security model, otherwise your server application is open to everybody. Building a web solution over a library like Indy, and without much server-side development support, lead only to aborted attempts unsuited for the large scale deployments the web wave allowed and needed. The web application is not only client-side development - there is a lot of server-side code, something Delphi refuses to understand. Meanwhile Microsoft leveraging the new server capabilities of its OS, was able to build - maybe slowly, true, but eventually it got there - solutions to deliver the whole stack of applications - from servers to clients.

Meanwhile BorInCodeDero instead of asking why more and more Windows developers was abandoning Delphi and swicthed to other tools, thought they needed to target different platforms. Again, it targeted Linux the wrong way - guess what? Desktop applications! Again, blinded by fashion, it didn't undestand most Linux machines were (and are) servers, not desktops. Kylix was a failure, and couldn't have been different. The OSX/mobile approach still fits their narrow view that software development means "desktop end-user application" only.

Sure, there is still Datasnap... the only real attempt BorInCodeDero made at server side development. The old DCOM one was introduced in D3, update with code borrowed from Dan Miser's in D4, and then never improved. It still implements DCOM as defined in Windows 95/NT4. There was ever nothing to help you writing MS-RPC applications. The "new" Datasnap was designed totally overlooking security (while Delphi still lacks VCL libraries for CryptoAPI support) - as if news about security incidents never reached BorInCodeDero - and manageability - as if the actual networks were still the small LANs of the '90s. SOAP support has been updated for clients only (and many SOAP features are still unsupported), sure, the latest fashionable buzzwords, "REST" and "JSON" have an implementation.... are they enough?

Thereby, we have a large part of the envelope Delphi can barely cover. But even in the part that was its strongest support, desktop development, Delphi lost a lot of ground, while chasing butterflies on other platforms. Many new features introduced from Windows 2000 onwards are still outside Delphi support, and needs third party libraries. Don't get me wrong, I like a good third-party ecosystem - but relying on a lot of third party code - a lot of which became no or barely supported later, may be a risk. I already wrote about CryptoAPI, but I can add many more. For example, Delphi has no support for the Window Event Log. It added Control Panel applet support, but when those were replaced by MMC plug-ins, BorInCodeDero "forgot" to add support for them. It took ages to add full theme support for all VCL controls. Still, many new features introduced with Windows 7 are still unsupported. WinRT support is not on the horizon. And I could add more and more.

And what happened? Most Delphi developers, well, became Delphi developers. Too many of them develop applications only inside the envelope provided them by Delphi - and often still adopting habits that should have been forgotten since Windows NT4 or 2000 - and don't write applications exploiting Windows fully - applications just floating over the surface of Windows and never well integrated and rooted inside of it - even if they have no "cross platform" need and would really benefit from better use of Windows. I see them. If they have to write to a file, [they still use (http://wiert.me/2014/03/10/delphiturbo-pascal-appendwithretry-for-text-files-to-retry-append/)](http://wiert.me/2014/03/10/delphiturbo-pascal-appendwithretry-for-text-files-to-retry-append/) the old, deprecated, dreadful, non thread safe Pascal file API (the Assign/Rewrite/Reset/Append one). Believe me, it's bad even for logs. Need a great logging facility? Look at [Windows Event Tracing. (http://msdn.microsoft.com/en-us/library/windows/desktop/bb968803(v=vs.85).aspx)](http://msdn.microsoft.com/en-us/library/windows/desktop/bb968803(v=vs.85).aspx) It's there since Windows 2000...

And as BorInCodeDero did for its pseudo-tile support, they will set up a TCP/IP server - or even an HTTP one - to have two Windows processes on the same machine talk to one another. Better, faster, more secure ways of communication are unknown to them, because they have no other little squares in the component palette to drop to a form. Slowly, developers using Delphi to develop Windows applications, turned into developers developing Delphi applications only. Twenty years ago, my colleagues had a good understanding of Windows development, and could code great Delphi apps. Today, most Delphi developers left I know, have a very limited knowledge of Windows - and not only because Windows became a so huge operating system. "Sandboxed" inside Delphi, they lost touch with the OS, and Delphi OS support became much narrower than it used to be.

Thereby ask yourself... are you still a Windows developer, or only a Delphi one?