

borland.public.delphi.non-technical: Re: My longest lived program retired after 22 years.

## Re: My longest lived program retired after 22 years.

**Source:** <http://coding.derkeiler.com/Archive/Delphi/borland.public.delphi.non-technical/2004-06/3130.html>

---

**From:** Dennis Landi (*nada\_at\_nada.com*)

**Date:** 06/21/04

Date: Mon, 21 Jun 2004 07:36:19 -0400

John Herbster's Question:

- > *Have you ever wondered about how to build a long lived*
- > *programs with modern tools? Which present day technologies*
- > *do we need to use for programs intended to last 25 years*
- > *with minimal maintenance? And what can we do we plan for*
- > *the transition at their eventual replacement?*

-----

"John Kaster (Borland)" <johnk@borland.com> wrote in message  
news:40d664e6@newsgroups.borland.com...

- > *Dennis Landi in <40d633f5\$1@newsgroups.borland.com> wrote:*
- >
- > > *Are you addressing John's question? I don't believe you are, but I*
- > > *was.*
- >

- > *If you care to spell out how your concern for a 64-bit Delphi compiler*
- > *(support for one future platform) relates to John's question, I'm sure*
- > *people here can explain how there are surer ways to guarantee the*
- > *compatibility of your Delphi code for future computing platforms.*
- >

Assuming you have actually asked a sincere question I will attempt a sincere answer.

Its very simple, and I can illustrate with my own current dilemma.

Scenario: It has become evident that there is and excellent chance that Borland will not upgrade its Delphi product based on the Win32 platform with a Delphi version based on the identical Win64 platform, (despite mine and few others best efforts to raise awareness on the issue). Furthermore the only language that I see as secure against the threat of the whims of a single proprietary vendor; while enjoying universal penetration on all platforms is C++. Now, my recent survey of state of the art C++ culture (by surveying the leading authors in the community) suggests that it may take me

Re: My longest lived program retired after 22 years.

borland.public.delphi.non-technical: Re: My longest lived program retired after 22 years.

up to two years to fully master C++ to the same level I am currently at with Delphi. This is an important data point.

Nevertheless, I am at a point where I need to reassess my own plans. There are some new products that I want to release in a few months. Based on on-going preparation over preceding years, some products will require "minimal" effort and I can see the benefits of truly planning their obsolescence within 5 years depending on the uptake of Longhorn and my own mastery of C++. These short-lived products and will have easily justified their existence with the revenue they have generated during that time, and hopefully it will be enough to justify their rewrite in C++ in 3 to 5 years from now, if that makes sense to do so at that point in time.

But... I have another major product for which I had planned a start on development. I estimate it will take me two years to build, with a planned life-time of 10 to 15 years, and there is the rub. I don't see .NET as a suitable technology for this product as it requires optimal performance; nor am I ready to believe that Delphi.NET will absolutely be around 5 to 10 years from now. In terms of performance, I see Delphi or C++ as a suitable platform. But two years from now, Delphi/Win32, without a direct upgrade path to Win64 (note: the explicit and obvious 64-bit connection, JK !!), will be "deprecated" technology at that point. So I just don't see how it would make sense to start development now on a product with a technology that will be obsolete by the time I am done. That's truly a problem for me. I still haven't completely decided what to do. At this point, I see no alternative but to delay development on this product. The ACE C++ communications framework seems like a perfect replacement for Delphi Sockets frameworks; and I feel encouraged by this. But I still need some proof of concept work to build further confidence that I am on the right track. But, needless to say, all this directly addresses by John Herbster's question.

<<Have you ever wondered about how to build a long lived programs with modern tools?>>

Yes!!!!

<<Which present day technologies do we need to use for programs intended to last 25 years with minimal maintenance?>>

Good question!

C++ (based) technology looks like the only viable candidate on the horizon. It is relatively immune to the whims, blindness and mistakes of single vendors with control of a language who don't necessarily have the best interests of that language in mind when making business decisions. Bottom line, trusting one's future to a language that is OWNED by a single commercial entity is a BAD IDEA for long-term projects that require massive investment up-front and then are intended to exist as living/breathing products for a decade or longer.

Re: My longest lived program retired after 22 years.

borland.public.delphi.non-technical: Re: My longest lived program retired after 22 years.

<< And what can we do we plan for the transition at their eventual replacement?>>

Think ahead! Make your best efforts to see the future! If you are using a proprietary language OWNED by a single vendor, then make your best effort to get answers to questions on the future of that language. Does the vendor plan to expand the language's presence on hardware platforms, or shrink it? Or abandon it altogether? Is the vendor making logical moves with that language. I.E. Given a Win32-based Delphi, are they taking the next logical step and releasing Win64 based version? Or are they refusing to say anything? And why would that be? Keeping their options open? Hmm. Is that a good enough basis for your own business decisions?

>>*plan for the transition at their eventual replacement*

Yes, this is a must.

- 1) Ascertain the lay of the land. What is the estimated longevity of "Technology X".
- 2) At the point of obsolescence, make your best efforts to ascertain the projected state of the Technology Milieu at that future point in time.
- 3) Attempt to weigh in the balance maintaining (bandaging) deprecated technology against the cost/benefits of developing the new replacement.
- 4) Reiterate steps 1 -3, on an on-going basis. Never deny new facts as they emerge, but incorporate them into your planning.
- 5) At the end of the day, the software developer needs to assess to what extent his/her investments in time/energy/money/source-code has been protected.

Is your code-based expendable? To what extent? Surely its expendability is directly related to the investment of time/money/effort invested to create it. What are the threats to that investment? Is your codebase dependant upon the whims/decisions of a single vendor? This could be a problem. Attempt to get some answers from that vendor in regard to the longevity and security of your codebase. If you don't like the answers, an adjustment in plans may be in order.

Now, JK. I guess you can continue you rather unimaginative attacks on me as a Thread-jacker; and thereby entirely sidetracking the thread (ironically). Or you can just stay quiet, as you have now really nothing of worth to contribute to the discussion. Or you can make sincere attempt to address these points.

And btw, last time I checked you are a Borland representative, so I will assume you speak for Borland unless you insert the necessary caveats.

>*From my perspective Delphi/Win32 has about 3 years of solid life left in it as a vital platform. For projects within that time-frame many short-lived projects can be executed. After that? Who knows? Each developer will have*

Re: My longest lived program retired after 22 years.

borland.public.delphi.non-technical: Re: My longest lived program retired after 22 years.

to survey the technology landscape and make their best efforts to see the future.

JK, is there an appropriate V.P. at Borland I could forward this message to? What would the his/her email address be?

-d