

Re: teaching a child – console or GUI

Source: <http://coding.derkeiler.com/Archive/Delphi/comp.lang.pascal.delphi.misc/2004-08/0593.html>

From: J French (erewhon_at_nowhere.com)

Date: 07/28/04

Date: Wed, 28 Jul 2004 10:37:40 +0000 (UTC)

On Wed, 28 Jul 2004 08:52:49 +0000 (UTC), Marco van de Voort
<marcov@stack.nl> wrote:

>On 2004-07-27, J French <erewhon@nowhere.com> wrote:
>
>><marcov@stack.nl> wrote:
>>
>>>> *Could you replicate such an interface for your 'pet' ?*
>>>> *(ANS: undoubtedly)*
>>>
>>>*No. It doesn't perform. And it is not a pet.*
>
>
>> *Surely, you are dealing with a lot of historical data, so it is*
>> *possible to set up some pretty fancy 'accelerators'*
>>
>> *What bit of it does not perform*
>
>*Unfinished sentence on my part, I think. I think I meant that the*
>*application doesn't perform int 21h's, or any other legacy technique.*

I understood you meant 'perform' in the sense of work well enough
– eg: too slow

>> – *or rather what is it that makes it slow if you keep most of the data*
>> *on disk ?*
>
>*Filters/queries over 6 million objects (in 8 entities/tables) must be in an*
>*acceptable time. Multiple users might run queries at the same time, but not*
>*many (say 1-4 users)*

I can see that, however one can make a system 'learn'
eg: a simple query can store its results in a BitMap that it slaps to
disk under a file name that is ... well the query
It does not need to be bitmaps, sparse results (eg: all 2003 records)
can be just a list of 4 byte pointers

A place I once worked used to thrive on selling databases to financial institutions (they still do) and we developed a whole load of ways of accelerating searching and sorting

*>Also note that 6 million objects are a lot more lines in a RDBMS, because you
>have all kind of coupling tables for 1 to many relations.*

Are these objects very complex, or are they really a bunch of pointers

>The use of indexes is limited, or you really need a lot of indexes.

At its simplest an 'index' is only a list of sorted 4 byte pointers
One can hold a lot of those on disk ...

>
>>>> *Try it and kick the data set up to 8Gb and see what happens*
>>>
>>>*That happens in 2020. 300MB/year*
>>
>> *Will they be interested in 2001 data in 2020 ?*
>
>*No, 5 years max. And even that is unlikely. They probably could do with 2,3*
>*years real data, and global stats from the other 2,3 years. But it is not*
>*worthwhile to code that.*

They could just pull up the 5 years minus system from a DVD
I also 'purge' my files on my main system, in the knowledge that the clients have numerous archive data sets

*>However that 300MB/year figure and the five year figure is the
>current situation.*

Any chance of it going wild ?
It is some type of 'local government utility' system isn't it ?
No chance of them taking on another municipality ?

*>We didn't know the exact sizes and date ranges yet when we made the
>decision. At the time we were afraid of hitting the Delphi limit of 2–3 GB.
>It was more 780MB/year in the initial version, and application memory
>on top of that. Improved indexing, and packing some data decreased the
>size of the data.*

eg: trashing white space and tokenizing longer fields I guess

It sounds as if you have some raw files that you crunch and slap into RAM – rather like building a CD 'database'

>>>> *Try abstraction, if only to prove it does not work.*
>>>
>>>*Been there, done that, got the t-shirt, and it is already pale.*
>>

>> *Yes, well as we get older, we get craftier*

>> *Maybe it is time to look at it again*

>

>*I had to be convinced too. (by my colleague). But now I have done a few projects with the mentality,*

>*I wonder why I never saw it myself.*

You mean that you had to be convinced of the 'CD in RAM' approach ?

It kind of makes sense, but it /is/ possible to emulate that with very

little speed hit by making the App think that it is just using RAM,

while it is really using a large 'window' onto a file

After all, your current system must be paging memory in and out, even

if you have a heavily chipped up 'data server'

>*We are still thinking in a DOS way about memory I think. Memory as precious*

>*resource. It is a commodity now.*

You mean you are, or I am, or both of us ?

I agree it is a commodity, but if anything, that is the problem

It stops people looking at the underlying data structure

I mentioned my pal who works for DEC ... Compaq .. ?? now selling
extra RAM to banks to speed up jobs

However at the same time another friend of mine who I've lost contact
with made a very good living as an old time programmer making banks'
overnight batch jobs run faster – just a bit of knowledge of the order
of the data so that disk heads do not go scudding all over the place

I can, for example, see that if your data is what I think it is, then
you could have multiple copies of the relevant stuff sorted in
different order.

It would also be possible to split the data into parallel files that
reside on really inexpensive EEPROM disks

>> *Something like File Mapping ?*

>

>*Nope.*

It looks interesting to me, although I would prefer to control it
explicitly myself.

>> *Realistically I have things that should really be re-written, but I*

>> *defend my not doing so on the grounds of*

>> *– why upset it*

>> *– its working so the users will not benefit*

>> *– I'm idle*

>> *– I'm not interested*

>>

>> *I can seldom point to anything and say that it cannot be improved, or*

>> *rather 'future proofed'.*

comp.lang.pascal.delphi.misc: Re: teaching a child – console or GUI

*>I would not try to retrofit such system to a RDBMS based solution, unless
>you have severe (as in magnitudes) performance problems. The way one works
>is too different. It is more meant for new development.*

I really would not use a 3rd party RDBMS for anything (unless seriously well paid) however building ones own filing system is not particularly hard, and is rather interesting

Your system has rather caught my interest, probably because it sounds similar to problems I've worked on in the past.

I really do believe that the key to speed is algorithms, not RAM