

Re: Java is becoming the new Cobol

Source: <http://coding.derkeiler.com/Archive/Cobol/comp.lang.cobol/2008-01/msg00161.html>

- *From:* LX-i <lx0007@xxxxxxxxxxxx>
 - *Date:* Sun, 06 Jan 2008 20:45:29 -0700
-

Judson McClendon wrote:

"LX-i" <lx0007@xxxxxxxxxxxx> wrote:

And yes, I'm programming using OO every day, have been doing so for over a year, and I have still yet to see anything I couldn't have done simpler and easier as procedural.

I've been doing it right at 10 months (so I guess I'm a little behind you), but I'm of a completely different opinion. OO has saved me *loads* of time. In fact, now when I hear a problem being described, my mind automatically starts breaking it up into the objects that will be affected, and the information they need to exchange to make it happen.

Just this past week, I had to write some code for a process that was very similar to another one. What I did was look at what the old one did, look at what I needed, *and* look at future known requirements, and I was able to break the original object up into three objects, then subclass the second to make mine. I didn't have to write *any* of that code, because it had already been written. <snipped>

Had that code been already written procedural code, with comparably strict definition and usage attributes, you would have been in the same situation. It appears to me, from your and Pete's comments, that you may be unfamiliar with very good modular structured techniques. I have done all of what you describe above, many times, using well defined, modular procedural code.

I've seen places where the type of programming I'm doing now would fit nicely into the system I used to work on. Of course, getting permission to do it would be the tough part. It's hard to pitch a perceived benefit most directly visible to the developers. (The "increased reliability for the users concept" is hard to quantify. "So, Jim, how many problems didn't we have this week?")

And the underlying support system to get it done was far more streamlined and efficient than the vastly bloated, inefficient OO tools we use now.

Re: Java is becoming the new Cobol

What tools seem bloated and inefficient to you? (I'm not saying they aren't – I'm just curious as to what you're referring.)

I have always worked toward modular, reusable code, with very good result. IMO, it is using the principles of well designed modules, data hiding, and other long ago defined structured principles that you are benefitting from, not OO per se. As I've said, I have yet to see one thing done in OO that could not have been done as easily, or more easily, and far more efficiently, using good procedural techniques.

Sometimes, though, you have to do the best you can in the environment you've been given. In the Unisys 2200 environment using their Network Database Server (DMS), you get a schema work area depending on how you invoke the schema. Your choices are working-storage, linkage, or common-storage. (Common storage is a Unisys thing – all linked modules have access to this area of memory.) However, it not only copies the schema area into the program, but the "DMCA" (Database Management Communication Area – basically, the block of information that holds your connection state) is copied in too.

The problem is, you can't realistically bring it into linkage, because the 200+ 01-level record definitions would get copied in. So, you're left with working-storage, which makes it visible to your program only, or common-storage, which makes it visible to programs linked in. Either way, the design of it naturally lends itself to a tight coupling with the database. Unless subroutines are coded with the schema defined the exact same way, they don't have access to it (even though the thread has been established).

Now, that being said (the perfect set-up for the "but I just can't do it" excuse), it COULD be done. However, without a complete database abstraction layer, these limitations are still going to be there. So, you get these monolithic programs with two or three screenful's worth of copy statements. The data's wide open, because it has to be to work.

I would absolutely love to create a data abstraction layer for that system. :) But, getting all the program changed to use it, getting them tested, then getting them out the door make it not feasible.

<reminiscing>

There was one program that I broke up into nested subprograms. It was our batch routine validator, and it did column-by-column verification of inputs, ostensibly to catch errors without having the overhead of starting a batch run. I modified it to have a nested subprogram for each transaction code, then within these, I copied in the screen working-storage (which had meaningful names), and changed the validation to use the names. However, because of the whole schema thing, I had to keep paragraphs in the main program to be able to do data validation against the database. :(The structure change benefitted no one but me (well, and the other folks who got to maintain it), and if I hadn't been quick on some other code, I would not have had time to do it.

Ideally, though, each of those would have been its own separate module.

</reminiscing>

Sigh. All of this is merely of academic interest, though, for the world has well and truly followed the Pied Piper of OO, for better or worse. It does

Re: Java is becoming the new Cobol

work, and I'm using it. I just grieve that this over-complex, inefficient path has been chosen for us, when a better, simpler, much more efficient one was already in place, had it been used with wisdom and discipline.

I really don't see the excessive overhead. That's why I asked you what environment you're working in. But hey, we've got to have *something* for all these cores in our modern computers to do! ;)

--

~~~~~

~/\ \_ o ~ Live from Albuquerque, NM! ~

~ \_ ^ | ~ ~

~ ~ ~ ~ ~

~ Business E-mail ~ daniel @ "Business Website" below ~

~ Business Website ~ <http://www.djs-consulting.com> ~

~ Tech Blog ~ <http://www.djs-consulting.com/linux/blog> ~

~ Personal E-mail ~ "Personal Blog" as e-mail address ~

~ Personal Blog ~ <http://daniel.summershome.org> ~

~~~~~

GEEKCODE 3.12 GCS/IT d s-: + a C++ L++ E---- W++ N++ o? K- w\$!O M--

V PS+ PE++ Y? !PGP t+ 5? X+ R* tv b+ DI++ D+ G- e h----- r+++ z++++

"Who is more irrational? A man who believes in a God he doesn't see, or a man who's offended by a God he doesn't believe in?" – Brad Stine

.