

## Re: New "base document" available

---

*Source:* <http://coding.derkeiler.com/Archive/Cobol/comp.lang.cobol/2007-04/msg00109.html>

---

- *From:* "Pete Dashwood" <[dashwood@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:dashwood@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 6 Apr 2007 02:12:51 +1200
- 

"Gary" <[spam@xxxxxxx](mailto:spam@xxxxxxx)> wrote in message  
[news:fNWdnXx-J-cyV4nb4p2dnA@xxxxxxxxxxxxxxxxxxxx](mailto:news:fNWdnXx-J-cyV4nb4p2dnA@xxxxxxxxxxxxxxxxxxxx)

Roger While wrote:

Might I also suggest once again to the J4 that some consideration  
be done about retrieving command line parameters.  
How a about a FUNCTION COMMAND-LINE?

Roger

Now that the thread has died down, I thought I would toss in a few  
thoughts.

Several years ago a subject was how to implement WinAPI calls. My  
suggestion was to use the (Soon-to-be-Defunct) MCS (Message Control  
System) which was for passing messages, control, and data between the  
system, run units, and others.

I remember writing a test of it many years ago. It worked flawlessly. Queued  
messages on a disk that was accessible across the network (pretty cutting  
edge at the time). Although it was primitive by today's standard, it was  
simple, and it worked.

Most of which could be implementor defined. J4 also (in its wisdom),  
dropped the Asynchronous Processing Module (or whatever), and now other  
languages are discussing how to control threads. It seems that COBOL was  
ahead of the times, but is deleting components as/just before they are  
needed, leaving them to be implemented in other languages.

I'd not thought about that, but now you mention it... :-)

Re: New "base document" available

Retrofitting Objects in COBOL was is not very compatible in several implementations. Adding those (CMS,APM) lost capabilities (standards) may never happen again. They both ARE still running happily on several Unisys 2200 mainframes , located at several former Baby Bells. (Message queues,interrupts,system and user-to-user passoff, etc.. all implemented under the standards.)

Amazing! This is the first time I have ever heard of anyone implementing the MCS for live applications.

Other languages may be more suitable for certain types of processing, or so it has been said, but I think much could still be done with what HAS existed if not for short-sighted Working Groups.

I can't believe I'm saying this :-), but, it may not be entirely fair to blame the working groups. At least, not entirely... It seems to me that much of the problem has been the red tape, bureaucracy, Civil Service mentality and a general Management inability to recognise and cut through it, which has dogged J4 in the past. Reading Bill's excellent proposal, it looks like there are still problems, inasmuch as there simply aren't enough people volunteering to do work. Bill's paper will be controversial, but it is sensible and realistic. I hope they consider it seriously.

Pete (I think) posted a link to a video about Lambda expressions for data retrieval, which sound a lot like the navigational/network DBMS using sets, only without issuing an I/O until the final expression (ie object.object.object.xxx...).

Yes, it was I, and I am pretty excited about this stuff. It could be implemented with a network DB model but is being primarily targeted to relational, because that's where the market is... Lambdas are really pointers to functions that manipulate data. "Objects" if you like...The point of using them is that the "function" can do vast amounts of work using new forms of Query Expression which are "SQL-like" but far removed from SQL in form and implementation. By using a Lamda, the programmer can just continue on in his/her logic as if the data required had been presented back, without needing to get sidetracked by how to obtain it. It is simply requested, and arrives wherever the Lambda is used. (Bit like Magic, really... :-)). Lambdas can be nested and in effect a single query expression may carry out the equivalent of thousands of lines of SQL. The nice thing is that the system can analyse and split down the query expression, even actually running it on separate cores if available, and, as you noted, nothing is actually retrieved until the algorithm is complete. (Deferred execution). Too much to go into here, but I predict we'll be

Re: New "base document" available

seeing more of it in the next few years, especially as multicore systems become more commonplace.

This capability was present in the CODASYL specs for SETS, even the waiting for the final expression before issuing an I/O (Is M\$ reinventing CODASYL?).

I agree it is very similar to what you are describing, but I suspect the subtleties are different. I don't know anything about CODASYL SETS so I can't comment. I do know that it is important not to label something an ITSA, when it is actually an ITSLIKE :-)

As for MS re-inventing stuff, I think it's a pretty fair bet they would do that :-).

Sometimes an old idea can be updated and improved, using modern approaches and technologies.

The Mazda MX5 is a Japanese "re-invention" of the Lotus Elan.

Having driven both, the MX5 is a better motor car.

(The one I have, I have had for 16 years and have no intention of parting with it... It looks like it just came out of the showroom, and it gives me hours of delight and amusement...As it is sub-tropical where I live, I get to drive it with the hood down for around 5 months of the year.)

In the Sperry 1100 version we were told to call them "Objects" or "items" because it was the DBA's definitions that described the "item" which could be anything from a PIC x up to and including a record description. The DMR was actually an interpreter, reading the ASCII text of the COBOL description, exactly like the relational DBMS systems I have seen do when processing data "items" or stored-procedures...

That certainly sounds years ahead of its time...:-)  
<snip>

Pete.

.

Re: New "base document" available