

## Re: Infinite Loops and Explicit Exits

*Source:* <http://coding.derkeiler.com/Archive/Cobol/comp.lang.cobol/2004-11/0257.html>

---

*From:* Lueko Willms ([l.willms\\_at\\_jpberlin.de](mailto:l.willms_at_jpberlin.de))

*Date:* 11/04/04

Date: 04 Nov 2004 09:43:00 GMT

. On 02.11.04

wrote [charles.stevens@unisis.com](mailto:charles.stevens@unisis.com) (Chuck Stevens)  
on /COMP/LANG/COBOL  
in [cm8rqo\\$1mg\\$1@si05.rsvl.unisis.com](mailto:cm8rqo$1mg$1@si05.rsvl.unisis.com)  
about Re: Infinite Loops and Explicit Exits

LW>> So one can by and by change such a program into something  
LW>> readable and maintainable.

CS> Yes, you can. But do you wanna take on restructuring a 500,000-line  
CS> monolithic COBOL program to make that happen?

I would prefer to avoid such a task, and I guess anybody else also  
prefers not to touch this monster, and its fate will be to be replaced  
completely by a package bought or by a complete rewrite in another  
language and, probably, on another platform.

But I would guess that this half-million-line monster can be melted  
down to at most half of those lines, if not even more, and be speeded  
up at the same time.

CS> There are cases in which EXIT PERFORM is, I think, clearer than GO TO  
CS> <some-arbitrary-paragraph-somewhere>.

That's why it is in the current standard. But I took issue with your  
proposal to allow it to effect a statement like this:

```
MAIN SECTION.  
LOOK-AT-THIS.  
  PERFORM procedure-name-1 UNTIL condition-1  
  PERFORM some-other-procedure
```

where "procedure-name-1" is somewhere else in the source code, and  
might look like this:

```
A)  
  procedure-name-1 SECTION.
```

```
BEGIN.  
  do-something-here  
  IF condition-2  
  THEN  
    EXIT PERFORM  
  END-IF  
  do-something-else  
  .
```

instead of

```
B)  
procedure-name-1 SECTION.  
BEGIN.  
  do-something-here  
  IF condition-2  
  THEN  
    GOTO ENDE OF procedure-name-1  
  END-IF  
  do-something-else  
  .  
ENDE.  
EXIT.
```

But the EXIT PERFORM above does something completely different from the GOTO in the second example, or in the third example given here:

```
C)  
  
procedure-name-1 SECTION.  
BEGIN.  
  do-something-here  
  IF condition-2  
  THEN  
    EXIT SECTION  
  END-IF  
  do-something-else  
  .
```

B) and C) are actually functionally equivalent – when "condition-2" is TRUE, then "do-something-else" is not processed. According to the PERFORM in MAIN SECTION, LOOK-AT-THIS paragraph, the procedure-name-1 would be repeated, until condition-1 gets true.

But A) would cause something completely different: independently of the value of condition-1, it would break the loop and cause the program to continue with "PERFORM some-other-procedure".

And this, the possibility that the condition-1 which guards the repetition in the main loop statement is being overrun by some minor condition in the PERFORMed procedure is the additional obfuscation

which the current standard explicitly prohibits, and with good reason, and which your proposal would make possible.

That is why I object to it, and strongly.

BTW, it would be better to write instead of A), B) or C) this:

```
D)
procedure-name-1 SECTION.
BEGIN.
  do-something-here
  IF NOT condition-2
    do-something-else
  END-IF
.
```

Shorter, simpler, safer, easier to understand.

CS> Moreover, \*elegance\* of coding style should not be dictated by the  
CS> COBOL standard, particularly when the requirements of elegance  
CS> invalidate existing programs. Matters of elegance are best left to  
CS> the educators.

The programming language should provide the means for elegance.  
Beginning with COBOL-85, this is really possible by using COBOL.

Elegance comes with simplicity and appropriateness to the task.

Yours,  
Lüko Willms <http://www.willms-edv.de>  
/----- L.WILLMS@jpberlin.de --- Alle Rechte vorbehalten ---

Die gefährlichsten Unwahrheiten sind Wahrheiten mäßig entstellt. -G.C.Lichtenberg