What is the behavior difference of STOP THREAD?

Question:

I have heard that isCOBOL behaves differently on a STOP THREAD statement since a YIELD is needed. Please elaborate.

What do you mean that a YIELD is needed? What exactly is the difference in behavior? What do I need to change in my program that uses STOP THREAD? How else can one thread stop another (e.g. change STOP THREAD to SEND and do a RECEIVE in the other threads)? What is the workaround?

Do you plan on fixing the above behavior to match ACUCOBOL-GT?

Answer:

The main issue is that isCOBOL threads are real threads (java thread)and Java APIs don't allow running threads to be killed. (http://java.sun.com/j2se/1.5.0/docs/guide/misc/threadPrimitiveDeprecation.html)

Originally, STOP THREAD was not supported at all, and the only way to stop thread programatically was to change the COBOL code using messages. Since then, we have found a way to implement STOP THREAD. However, since isCOBOL uses the Java thread model, a call to the yield() method is required in order to allow thread interruption. For this reason we have added a new COBOL statement named YIELD to be used where the running thread must be stopped.

COBOL developers may need to control exactly where a running thread is stopped. STOP THREAD can be an unsafe statement to use because thread closing is not controlled. It is better practice to send a message closing the thread than to just to close the thread.

If you use STOP THREAD without also using YIELD statements, then the STOP THREAD would never return.

Programs using STOP THREAD should not execute a blocking ACCEPT, or other blocking statement, because this may cause the program to hang.

The solution is to use SEND to stop the thread, and then add ALLOWING MESSAGES FROM ANY THREAD to the ACCEPT statement. You could also have a timeout on ACCEPT or other statements, so the code can periodically call YIELD.

Online URL: https://support.veryant.com/phpkb/article.php?id=44