Why move to Free-Form RPG?

The free-form coding style has been available for RPG IV since IBM released V5R1 in the spring of 2001. Since that time, Linoma has used the free-form syntax extensively for in-house development. Based on positive feedback from our developers and well-known experts in the RPG community, we are highly recommending the RPG free-form syntax to our customers.

We believe that the free-form syntax is a big improvement to the RPG language and has many productivity advantages. Developers are finally no longer bound by the fixed columns of traditional RPG calculations. Now you can simply enter your RPG logic in a natural left-to-right fashion, similar to the coding style in other modern languages (i.e. Java and Visual Basic).

While the free-form syntax only is allowed in the area where C specifications are normally entered, it is definitely a huge step in the right direction.

To give you a better understanding of why we are recommending the free-form RPG style, we have compiled a list of benefits we believe it offers over traditional fixed-format syntax:

1. Nested logic can be indented
2. Source code can be entered faster
3. More room to enter long expressions
4. Right-hand comments can immediately follow the logic
5. Free-form will cohabitate with fixed format specs
6. Growing number of source examples only in free-form
7. IBM is focusing its resources on free-form RPG
8. Easier to learn for newcomers
9. Enhances your career

These benefits are detailed in the following pages.

If you elect to use the free-form style, any existing RPG IV development processes (ie change control) should remain unaffected since the source member type and program behavior will not change. IBM's RPG compiler is intelligent enough to determine the difference between free-form and fixed-format specifications, so you can continue to compile your RPG IV programs as you always have.

We believe there is only two reasons why you may not want to code in the free-form style. The first reason is if you have to deploy your programs to an OS/400 release lower than V5R1 since free-form RPG is only supported in V5R1 and higher. The second reason is if you depend on a field cross reference tool that does not recognize the free-form syntax yet (check with your vendor). Otherwise you will be well served by exploring this much-improved coding style.

Once you're convinced that the free-form RPG style is right for your organization, we recommend using our RPG Toolbox product to convert your existing RPG logic into this new style. The Toolbox can be downloaded at www.linomasoftware.com.

Besides bringing your source code up-to-date with the latest style and methods available, Linoma's RPG Toolbox will also help you easily learn the free-form syntax by simply reviewing your source code before and after the conversion. With Linoma's RPG Toolbox, you can additionally use our new SEU line commands and source code snippets to easily create, maintain and indent free-form RPG logic.

Free-Form Benefit # 1 – Nested logic can be indented

Understanding and maintaining nested logic can be very difficult in fixed-format RPG, especially when the logic is nested several levels deep. Trying to keep track of which logic is associated with which IF, ELSE and DO statements can be very time consuming. For example, try to quickly comprehend the logic in the following fixed-format example.

```
If      Test = 3
Eval    Work = A + B
If      Work = 1
Eval    A = A + 1
Else
If      Work = 2
Eval    B = B + 1
EndIf
EndIf
Eval    C = C - 1
Else
Return
EndIf
```

The above logic is hard to follow, even though it’s nested only 3 levels deep. Unfortunately its column-bound nature does not make it easy to trace the logic. You may be tempted to print this example out and use a pencil to draw lines between the IF, ELSE and ENDIF statements for a better understanding.

What if this same example was coded in the free-form RPG style:

```
If Test = 3;
    Work = A + B;
    If Work = 1;
        A = A + 1;
    Else;
        If Work = 2;
            B = B + 1;
        EndIf;
    EndIf;
    C = C - 1;
Else;
    Return;
EndIf;
```

As you should see from the above free-form example, the power of indenting source makes the free-form logic easier to understand than its fixed-format equivalent.

An RPG program will typically have a good deal of nested logic within it. By using the power of indenting nested logic within free-form RPG, your productivity will be greatly enhanced when analyzing and maintaining applications.

Free-Form Benefit # 2 – Source code can be entered faster

Instead of being concerned with entering operation codes and variables into the correct columns with fixed-format C specifications, the free-form style allows you to enter source in a natural left-to-right fashion. Just insert a new line and start keying your logic!

As an added bonus to the free-form style, you don't need to key in the EVAL or CALLP when assigning values or calling procedures. These operation codes are assumed.

Fixed-Format:

```
Eval     Work = A + B
CallP    Process(Work)
```

Free-Form:

```
Work = A + B;
Process(Work);
```

As an interesting exercise, we've added up the number keystrokes it takes to enter one EVAL statement using both styles. For instance, to enter an EVAL statement using a fixed-format C specification within IBM's SEU source editor, a developer will most likely enter the following keystrokes:

1) Key in IPCX on a sequence number and press Enter to prompt (5 keystrokes)
2) Tab to the operation code column (3 keystrokes)
3) Enter the word EVAL into the operation code column (4 keystrokes)
4) Tab to the extended factor 2 column (1 keystroke)
5) Key in the expression
6) Press the Enter key (1 keystroke)

Not considering the keystrokes to enter the expression itself, a single EVAL statement would require 14 keystrokes in fixed-format RPG.

What if you keyed the same expression in free-form RPG:

1) Key in an I on a sequence number and press Enter to insert a new line (2 keystrokes)
2) Key in the expression without the proceeding EVAL since it is assumed
3) Key in a semicolon to end the expression (1 keystroke)
4) Press the Enter key (1 keystroke)

Again, not considering the keystrokes to enter the expression itself, it would take only 4 keystrokes to enter the same line in the free-form style. Compared with the fixed-format example, you would save 10 keystrokes when entering this one line of logic.

Considering the hundreds of lines of code entered into a typical RPG program, the total keystrokes saved using the free-form style could be significant. Less keystrokes equals more productivity.

Free-Form Benefit # 3 – More room to enter long expressions

In fixed-format RPG, there are 45 characters available in the extended factor 2 area. If your expression extends beyond 45 characters in a C specification, then you have to create a new continuation line to key in the remainder. However using the free-form style, you can enter up to 73 characters of an expression per line.

Fixed-Format:

```
Eval     MailLabel = Name + Address1 + Address2 +
         City + State + Zip
```

Free-Form:

```
MailLabel = Name + Address1 + Address2 + City + State + Zip;
```

By having more room for long expressions in the free-form coding style, the number of lines needed in your RPG program will be reduced and your expressions will be easier to read and maintain.

Free-Form Benefit # 4 – Right-hand comments can immediately follow the logic

Using fixed-format RPG, right-hand comments must start in position 80. If your source file is 112 bytes in length, then you have only 20 characters of comment area. This narrow comment area will often force you to greatly abbreviate your comment text. Example:

```
Eval     Work = A + B                                     rpt. total NE+IA
```

Additional problems with traditional right-hand comments are:

1) You have to waste keystrokes in tabbing to the correct column to enter the comment
2) If your display session is set to a width of 80 columns, then you can't see the right-hand comments unless you window to the right
3) The excessive blank space between your logic and the right-hand comment may make it difficult to associate the two together (depending on the quality of your eyesight)

These problems can be eliminated when coding in the free-form style since you can immediately follow your logic with comments using the // notation (this is the same notation used in Java). Notice below how much clearer the unabbreviated free-form comment is.

```
Work = A + B;    // report total for Nebraska and Iowa
```

Since comments can be appended to your free-form logic, then more comment text can be entered. Free-form comments will also be visible within 80 column screens (if you don't go too far) and they will be easier to associate with your logic.

Instead of inserting your detailed comments above your logic, this "same-line" approach for free-form comments will keep your source at a reasonable number of lines while making it just as readable and maintainable.

Free-Form Benefit # 5 – Free-form will cohabitate with fixed format specs

While it's generally recommended to have all of your calculation logic in either all fixed-format or all free-form style (for consistency and readability), the RPG compiler will allow you to easily intermix the two styles within the same program. You simply need to start your free-form logic with the /FREE tag and end it with the /END-FREE tag. Example:

```
C                    MOVEA     WORKA         WORKB
 /FREE
  A = A + 1;
  B = B - 1;
 /END-FREE
```

Since almost every operation in fixed-format RPG has an equivalent free-form operation or built-in-function, then undoubtedly we believe 99+% of your calculation logic can be in the free-form style. The exceptions include only a handful of unique operations, such as the MOVEA (move array) operation. We're confident that IBM will address these few remaining fixed-format operations in future releases.

Free-Form Benefit # 6 – Growing number of source examples only in Free-Form

If you're similar to most RPG developers, you've learned a portion of your RPG techniques by reviewing code examples in IBM manuals, independent books and articles.

Since free-form has been available for a couple years now (as of this writing), most of the RPG authors are now writing their books and articles in this style. You will also begin to notice that more-and-more downloadable RPG source code is in the free-form style.

If you don't understand free-form RPG, then it will be difficult for you to take advantage these new RPG examples and downloadable source code.

Free-Form Benefit # 7 – IBM is focusing its resources on Free-form RPG

IBM is attempting to shed the iSeries stereotype as a legacy machine. This is evidenced by the somewhat recent iSeries support of WebSphere, Java and other modern technologies. Since RPG is the predominant 3GL language on the iSeries, we believe IBM thinks the free-form RPG syntax will also help give the iSeries a more positive image in the I.T. industry.

To modernize applications on the iSeries and to keep skeptical customers from moving to another platform, we believe IBM is eager for more developers to move into the modern free-form RPG style.

In V5R2, you can now retrieve records from a file using multiple keys without using a KLIST (key list). You can simply enter the multiple key fields within the CHAIN or READE operation. However this feature is only available in the free-form style. We believe that IBM will continue to add new techniques and operations, as illustrated in this example, which only work in the free-form style.

Free-Form Benefit # 8 – Easier to learn for newcomers

Most schools are training their students in free-form languages such as C, Java and Visual Basic. If you would like to bring junior-level developers into your shop, they will probably not be very excited about coding in a restrictive column-oriented language. The free-form style will help RPG trainees be more comfortable with the language since it will closely resemble the syntax they learned in school.

On the other side of this equation, if experienced RPG developers are also performing some Java development, even hard-core fixed-format RPG developers will find it more productive to switch back and forth between the Java syntax and the free-form RPG style.

Free-Form Benefit # 9 – Enhances your career

Besides the personal satisfaction of learning new technology, it is also important to remember that accumulating new skills will help ensure a long-term career in I.T. Failure to keep skills updated can lead to a static or downward career path.   For existing RPG IV developers, the free-form style will only take a few days to learn. It's an excellent investment of time, considering the amount of return you'll receive.