# RPG — A Sensible Alternative

*Steve Wright*

## PREFACE

The main purpose of this talk is not to present evidence of one programming language being better or superior to another. The decision of primary language is most likely already been made in your operation. This talk, however, is designed to make the statement that RPG is being used successfully in the HP world and should not be ignored becuase of dominate usage of any other language. Also, I want to present some uses that you may not have considered. Please bear with me on some elementary topics. But, I feel that some users have not been exposed to them, and hopefully will be beneficial.

## FACTS (AND HISTORY)

Consider the following:

- The most common programming language by far is not RPG.
- RPG to COBOL conversions are commonplace and packages that will perform 99% of the work are available from several vendors.
- Several former RPG users were told to switch to COBOL to get the most out of HP machines (mainly with support considerations).
- The person next to you at lunch today sneered at you when he heard that you used the RPG compiler.

## WHY DO WE INSIST ON USING SUCH AN ANIMAL?

Consider the following:

- COBOl shops are experiencing low productivity levels and are seeing report writers as an escape route.
- Operations that escaped the IBM System 3 world with excellant track records are using HP equipment to enhance a thing that is already good.
- Managers are finding out that matching programming languages with the assigned task can be a rewarding adventure.

My own personnel experience with Hewlett Packard equipment and available programming talent has led to the following situation. I have 90% of my programs in RPG, 8% In COBOL and 2% In FORTRAN. I am using FORTRAN In heavy math oriented problems, COBOL for mass data entry programs and RPG for all batch and Quicky Del routines. The heavy terminal usage programs are In COBOL and FORTRAN, and therefore are the heaviest used. But the RPG programs that support edits, update and reporting are the real workhorses that support every system.

My reasons for using RPG in so many batch programs is the speed at which the programs can be written, tested and Implemented. The run time difference between RPG and COBOL in batch routines for the most part has not been substantial and the implementation schedule of entire systems can be speeded up. Batch is not a daily task for the most part and therefore can be paid for in machine cycles instead of programming dollars.

Most programming managers have grown to love compiler languages with the ability to go between machine lines with minimal difficulty. The report writers of today are very impressive. I have looked very carefully at one very good one, and I still may ask Santa Claus for one next year, but if I were a small shop with limited programming resources (salary dollars not talent), I would have to consider RPG for everything from batch to data entry programs.

## HOW I DO IT

Below are some tricks of the trade that I use effectively to enhance my operation. Again, some of these items are elementary to many who will read this paper, but I find most people will find one or two things that will be new to them.

RPG supports only one-dimentionial arrays. I use some algorithms to make two-dimentionial arrays work. Suppose I wanted an array of 12 years data on the total of 6 product lines. To define the array, specify 72 entries (execution time array). To load the entries use X=year to load, Y=Product Code (1-6). The following code will locate the element.

```
*
*        X      SUB  1     X
*        X      MULT 12    X
*        X      ADD  Y     X
*    USE THE X ELEMENT OF THE ARRAY
*
* TO FIND THE MEANING OF THE ELEMENT SPECIFIED BY THE
* VARIABLE "WORK", USER THE FOLLOWING:
*
```

```
*  WORK   DIV  12   X  (NOT HALF ADJUST)
*  X      MULT 12   Y
*  X      ADD  1    X
*  WORK   SUB  Y    Y
*
*  THUS X= YEAR LOADED  Y= PRODUCT CODE
*
```

You may choose to print the results out using a 6 element array running down the loaded 72 element array by element and printing when you have filled up the 6 element array. I use just such a routine to look at the past 10 years history for product trends.

### R.A.F. (Random Access Files) (Addrout)

HP has released "Xsort" that will sort only using the key fields and the relative record number and dropping the large data portions of the record and leaving a file that contains only the relative record number of the records in the file arrainged in the order the file would be in if the entire record was carried along. The scheme allows very fast sorting of very large data files. This discussion is fully explained in the communicator #26. Use it. It can be a life saver and a hero maker. In case this all sounds familiar, It is the System 3 Addrout processing. Also, in processing files by random access (no keys) do not be afraid to declare a MPE file file as input chained and simply read it by relative record number. I do it all of the time to position myself back and forth within the file. "Chained" does not always mean "keyed."

### DSPLY

When using the DSPLY command, the limit of 8 characters for a constant in factor one is annoying. I use a compile time array at the end of the program to detail my prompts, giving me all of the characters I need without many cumbersome moves. Also keep in mind that you can send escape sequences in the DSPLY command to manipulate the terminal as well as ring a few bells. Very fast data entry programs can be written using DSPLY. I wrote one data entry program using DSPLY in two hours thinking that it was going to be a one-time program. It is still in use two years later with no changes.

### SETLL

The "SETLL" command is very useful in either KSAM or IMAGE file. Everywhere you have a user screaming for a name search routine, use the name as a duplicate key (or automatic data set) and use processing limit (SETLL) and the read command to give amazing results. The following code is an example:

```
*             ASKNM     TAG
*            'NAME?'    DSPLY TERM       NAME
*            NAME       SETLLMASTER
*            LOOPER     TAG
*                       READ MASTER              99
*       99               GOTO ASKNM
*            FULLNM     DSPLYTERM
*            'CONTINUE'DSPLYTERM         ANSWER
*            ANSWER     COMP 'Y'                 98
*        98             GOTO LOOPER
*              'STOPPED'DSPLYTERM
*                       GOTO ASKNM
*
```

## "LET ME CALL YOU SPL — (OR COBOL OR FORTRAN)"

The HP RPG compiler allows exits (calls) to external routines that can written in other languages. If you feel this is needed, keep in mind that it is available. The Orlando swap tape has such routines (such as calls to system intrinsics) can be very useful.

## INTERACTIVE WITHIN THE PRODUCT LINE

RPG and V/3000 is not a bad combination. The main reason I used COBOL with DEL was the ability of COBOL to read only one field at a time (a real time saver). However, the main thrust of V/3000 is in reading the entire screen at once and allow the V/3000 routines handle screen painting and repainting. (It does a pretty slick job at times.) RPG looks like a natural for usage with VIEW. V/3000 uses more screen dependent controls that insulate the programmer from the messy calls, that one should try it in RPG.

## "HOW ABOUT A DATE GOOD LOOKING?"

Communicator #24 tells of how the RPG programmer can specify "F" in column 17 of the header spec to

allow him/her to specify the sysdate from other sources than the system clock. One may use a disc file, or request the user enter the date as he runs the program. Also, the time2 command is explained in the #24 communicator. This command will return to the user the date in the format of "THU, JAN 10, 1980 9:25 AM JULIAN:010." This can be very useful in that the compiler allows you to select which fields in the above format you want to see, thus reducing the moves that would normally be associated with it.

## "CHECK PLEASE" (MOVEA)

I have a rather classy check protection routine that I am contributing to the swap tape. Use it in good health. The routine uses "MOVEA" extensively. The "MOVEA" usage is worth going through with beginner programmer types and intermediate types who have not been exposed to it. The full usage of arrays can enhance their productivity.

## CIRCLES AND CYCLES

The RPG programmer whether beginner or advanced, must, must, must, must know the RPG cycle. If he/she is not taught early, you simply have a COBOL or Assembler programmer with very restricted limits. The main attraction of the language is having the cycle do all of the grunt work for you. If you still think that total time processing occurs after a break instead of before a break, (there is a differance) you need to spend some serious time with the cycle flowchart in your HP or IBM Reference Manual.

## "WHERE DO WE GO FROM HERE??"

RPG programmers are a sturdy lot, but with the advantages of an international user group at our disposal, we should be doing more in the area of sharing ideas. At the San Jose meeting, several RPG users wanted to get together and start a special interest group, or a newsletter. The interest was high but no one kept the momentum going. I am open for suggestions for ways to start such a group. Some suggestions I have heard for the users are as follows:

- Have a newsletter with shared ideas as the emphasis
- Start a special interest group for addressing HP
- Set up a network of "help-lines" for RPG users
- Have special user group meeting before or after the international user group meeting each year.

One other item that I think may help is using some of the system 3, 34, 38 aids. That includes getting free subscriptions to such publications as *Small Systems World*. It is a monthly publication that has been well accepted by many small systems users. If you want the address to ask for the free subscription, please contact me.