BREXX/370 Callable External Functions

Mike Grossman & Peter Jacob

2. November 2020

With the new External Function feature, you can call compiled programs written in conventional language, as PL1, Assembler, and maybe more.
We closely adapted IBM's TSO/E REXX programming services:
https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.ikja300/progsrv.htm

How it works:

BREXX Call an external Program

To call an external program, you call it in the same way as a normal BREXX function:

```
say load-module(argument-1,argument-2,…,argument-15)
```

you can pass up to 15 arguments to the external function. The size of the return value can be up to 1024 bytes.

Example

```
Say RXPI()
```

RXPI is a load module which must be accessible within the link list chain. It does not have any arguments.

BREXX Programming Services

BREXX provides control blocks containing the arguments and a 1024 bytes return buffer.

Called Program

The program needs to match the BREXX calling conventions to manage the argument and return value handling. To ease it, we have isolated communication control blocks and internal functions in a copybook. Once included, it will transparently provide the functionality to the program.
Example, PI calculation:

```
RXPI:    PROCEDURE(EFPL_PTR) OPTIONS(MAIN);                            00000101
  %INCLUDE RXCOMM;                                                      00000201
…
```

Benefits

The performance of a compiled program is much higher than in BREXX. So if you have complex mathematical calculations, they will be significantly faster than code implemented in BREXX.
In our testing, we implemented an algorithm for calculating PI with 500 digits. In comparison, it was over 600 times faster than the same algorithm implemented in BREXX.

Example, PI calculation:

```
RXPI:    PROCEDURE(EFPL_PTR) OPTIONS(MAIN);                            00000101
  %INCLUDE RXCOMM;                                                      00000201
 /* -------------------------------------------------------------------00000301
  * CALCULATE PI USING THE ALGORITHM OF S. RABINOWICZ AND S. WAGON      00000413
  * INPUT VARIABLES                                                     00000501
  *   ARGNUM      CONTAINS NUMBER OF PROVIDED ARGUMENTS (MAX 15)        00000601
  *   ARG(I)      CONTAINS CONTENTS OF ARGUMENT I (1 TO 15)             00000701
  *   ARG_LEN(I)  CONTAINS LENGTH   OF ARGUMENT I (1 TO 15)             00000801
  * RETURN VARIABLES                                                    00000901
  *   RESULT      CONTAINS RESULT TO BE RETURNED TO BREXX               00001001
  *               THE RETURN VALUE MUST NOT EXCEED 1024 BYTES           00001112
  *   RESULT_LEN  CONTAINS LENGTH OF RETURNED STRING                    00001201
  * --------------------------------------------------------------------00001301
  */                                                                    00001401
  DCL (N, LEN) FIXED BINARY;                                            00001505
  DCL PI CHAR(512) VARYING;                                             00001611
  DCL TEMPI CHAR(9);                                                    00001709
  DCL PREDIGIT BIN FIXED(15);                                           00001806
  N = 500;                                                              00001912
  LEN = 10*N / 3;                                                       00002012
  PI='';                                                                00002112
 BEGIN;                                                                 00002212
  DECLARE ( I, J, K, Q, NINES) BIN FIXED(15);                           00002312
  DECLARE X FIXED BINARY (31);                                          00002412
  DECLARE A(LEN) FIXED BINARY (31);                                     00002512
                                                                        00002600
  A = 2; /* START WITH 2S */                                            00002712
  NINES, PREDIGIT =0; /* FIRST PREDIGIT IS A 0 */                       00002812
  DO J = 1 TO N;                                                        00002912
     Q = 0;                                                             00003012
     DO I = LEN TO 1 BY -1; /* WORK BACKWARDS */                        00003112
        X = 10*A(I) + Q*I;                                              00003212
        A(I) = MOD (X, (2*I-1));                                        00003312
        Q = X / (2*I-1);                                                00003412
     END;                                                               00003512
     A(1) = MOD(Q, 10); Q = Q / 10;                                     00003612
     IF Q = 9 THEN NINES = NINES + 1;                                   00003712
     ELSE IF Q = 10 THEN DO;                                            00003812
        TEMPI=PREDIGIT+1;                                               00003912
        PI=PI||SUBSTR(TEMPI,9,1);                                       00004012
        DO K = 1 TO NINES;                                              00004112
           PI=PI||'0';                                                  00004212
        END;                                                            00004312
        PREDIGIT=0;                                                     00004412
        NINES = 0;                                                      00004512
     END;                                                               00004612
     ELSE DO;                                                           00004712
        TEMPI=PREDIGIT;                                                 00004812
        PI=PI||SUBSTR(TEMPI,9,1);                                       00004912
        PREDIGIT = Q;                                                   00005012
        DO K = 1 TO NINES;                                              00005112
           PI=PI||'9';                                                  00005212
        END;                                                            00005312
        NINES = 0;                                                      00005412
     END;                                                               00005512
   END;                                                                 00005610
 END ; /* END BEGIN */                                                  00005712
   TEMPI=PREDIGIT;                                                      00005812
   PI=PI||SUBSTR(TEMPI,9,1);                                            00005910
 RESULT='3.'||SUBSTR(PI,3);                                             00006012
 RESULT_LEN=LENGTH(PI);                                                 00006112
 END RXPI;                                                              00006201
```

BREXX Version of the PI calculation program:

```
/* --------------------------------------------------------------------
 * PI USING THE ALGORITHM OF S. RABINOWICZ AND S. WAGON
 * --------------------------------------------------------------------
 */
RXPIR:
  N = 500
  LEN = (10*N/3)%1
  A.=2
  NINES=0
  PREDIGIT = 0 /* FIRST PREDIGIT IS A 0 */
  DO J = 1 TO N
     Q = 0
     DO I = LEN TO 1 BY -1 /* WORK BACKWARDS */
        X = INT(10*A.I + Q*I)
        A.I = INT(X//(2*I-1))
        Q = X%(2*I-1)
     END
     A.1 = (Q//10)%1
     Q = Q % 10
     IF Q = 9 THEN NINES = NINES + 1
     ELSE IF Q = 10 THEN DO
        PI=PI||PREDIGIT+1
        PI=PI||COPIES('0',NINES)
        PREDIGIT= 0
        NINES = 0
     END
     ELSE DO
        PI=PI||PREDIGIT
        PREDIGIT = Q
        PI=PI||COPIES('9',NINES)
        NINES = 0
     END
  END
  PI=PI||PREDIGIT
RETURN '3.'SUBSTR(PI,3)
```

Comparison of both implementations

```
PL1 Program:
3.141592653589793238462643383279502884197169399375105820974944592307816406286208
8998628034825342117067982148086513282306647093844609550582231725359408128481117
4502841027019385211055596446229489549303819644288109756659334461284756482337867
8316527120190914564856692346034861045432664821339360726024914127372458700660631
5588174881520920962829254091715364367892590360011330530548820466521384146951941
5116094330572703657595919530921861173819326117931051185480744623799627495673518
8575272489122793818301194912
Elapsed Time  0.49016099452972417 seconds
BREXX Program:
3.141592653589793238462643383279502884197169399375105820974944592307816406286208
8998628034825342117067982148086513282306647093844609550582231725359408128481117
4502841027019385211055596446229489549303819644288109756659334461284756482337867
8316527120190914564856692346034861045432664821339360726024914127372458700660631
5588174881520920962829254091715364367892590360011330530548820466521384146951941
5116094330572703657595919530921861173819326117931051185480744623799627495673518
8575272489122793818301194912
Elapsed Time  300.3606059551243 seconds
```

For the hardcore programmer

The current content of the Communication Interface follows. The long-winded coding is caused by the functionality of the old PL1-360-F compiler.

```
 /* --------------------------------------------------------------------00000500
  * REXX INTERFACE BLOCK  EFPL                                          00000600
  * --------------------------------------------------------------------00000700
  */                                                                    00000800
   DCL EFPL_PTR PTR;                                                    00000900
   DCL 1 EFPL BASED(EFPL_PTR),                                           00001000
         2 EFPLCOM  FIXED BIN(31),                                       00001100
         2 EFPLBARG FIXED BIN(31),                                       00001200
         2 EFPLEARG FIXED BIN(31),                                       00001300
         2 EFPLFB   FIXED BIN(31),                                       00001400
         2 EFPLARG  PTR,                                                 00001500
         2 EFPLEVAL PTR;                                                 00001600
/* ---------------------------------------------------------------------00001700
  * ARGTABLE ENTRIES AND RELATED DEFINITIONS                             00001800
  * ---------------------------------------------------------------------00001900
  */                                                                     00002000
   DCL EFPLARG_PTR      PTR;                                             00002100
   EFPLARG_PTR        = EFPLARG;                                         00002200
   DCL 1 ARGTABLE BASED(EFPLARG_PTR),                                    00002300
         2 ARGTABLE_ENTRY(15),                                           00002402
           3 ARGSTRING_PTR    PTR,                                       00002500
           3 ARGSTRING_LENGTH FIXED BIN(31);                             00002600
                                                                        00002704
   DCL ARGNUM      BIN FIXED(31);                                        00002810
   DCL ARG_LEN(15) BIN FIXED(31);                                        00002908
   DCL ARG(15)     CHAR(255) VARYING;                                    00003010
                                                                        00003104
   DCL ARG_PTR PTR;                                                      00003200
   DCL ARGSTRING CHAR(255) BASED(ARG_PTR);                               00003300
/* ---------------------------------------------------------------------00003400
  * EVALUATION BLOCK: EVALBLOCK                                          00003500
  * ---------------------------------------------------------------------00003600
  */                                                                     00003700
   DCL EFPLEVAL_ADR_PTR PTR;                                             00003800
   DCL EFPLEVAL_PTR     PTR;                                             00003900
   EFPLEVAL_ADR_PTR     = EFPLEVAL;                                      00004000
                                                                        00004104
   DCL 1 EVALBLOCK_ADR BASED(EFPLEVAL_ADR_PTR),                          00004200
         2 EFPLEVAL_ADR PTR;                                             00004300
                                                                        00004400
   EFPLEVAL_PTR = EFPLEVAL_ADR;                                          00004500
                                                                        00004600
   DCL 1 EVALBLOCK BASED(EFPLEVAL_PTR),                                  00004700
         2 EVALBLOCK_EVPAD1 FIXED BIN(31),                               00004800
         2 EVALBLOCK_EVSIZE FIXED BIN(31),                               00004900
         2 EVALBLOCK_EVLEN  FIXED BIN(31),                               00005000
         2 EVALBLOCK_EVPAD2 FIXED BIN(31),                               00005100
         2 EVALBLOCK_EVDATA CHAR(256);                                   00005200
                                                                        00005300
   DCL EVDATA_PTR PTR;                                                   00005400
   DCL EVDATLN_PTR PTR;                                                  00005500
                                                                        00005600
   EVDATA_PTR = ADDR(EVALBLOCK_EVDATA);                                  00005700
   EVDATLN_PTR = ADDR(EVALBLOCK_EVLEN);                                  00005800
                                                                        00005900
   DCL RESULT CHAR(1024) BASED (EVDATA_PTR);                             00006009
   DCL RESULT_LEN    BIN FIXED(31) BASED(EVDATLN_PTR);                   00006100
                                                                        00006200
   RESULT_LEN = 1;                                                       00006300
/* ---------------------------------------------------------------------00006400
  * COPY BREXX PARMS INTO PL1 STRUCTURE                                  00006503
  * ---------------------------------------------------------------------00006600
  */                                                                     00006703
   DCL AI BIN FIXED(31);                                                 00006804
                                                                        00006904
   DO AI=1 TO 15 ;                                                       00007004
      ARG_PTR = ARGSTRING_PTR(AI);                                       00007103
```

```
      ARG_LEN(AI) = ARGSTRING_LENGTH(AI);                               00007208
      IF ARG_LEN(AI)<=0 THEN ARG(AI)='';                                00007308
      ELSE DO                                                           00007403
         ARG(AI) = SUBSTR(ARGSTRING,1,ARG_LEN(AI));                     00007508
         ARGNUM=AI;                                                     00007605
      END;                                                              00007705
   END;                                                                 00007803
```