Session 9658

Introducing
LE Callable Services

Thomas Petrolino

IBM Poughkeepsie

tapetro@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

Language Environment

z/OS

z/VM

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

 * All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Agenda

- Overview
- Math functions
- Outputting messages
- Collecting error information.
- Manipulating date and time information
- Summary
- Appendix

Overview

- **Language Environment provides a set of callable services that can be called by programs compiled with a Language Environment conforming compiler.**
  - This includes assembler programs which use the CEEENTRY macro.
    - Fortran must use the AFHCEEN or AFHCEEF interfaces.

Overview

- Callable services beginning with CEE are valid on any platform.

- Callable services beginning with CEE3 are valid only on z/OS. (Yeah z/VM too)

Overview...

- Callable services may be called from AMODE 24 programs.
  - Language Environment will automatically switch to AMODE 31.
    - This may cause additional overhead with repeated calls.
    - Run AMODE 31 to avoid unnecessary switching.

Overview...

- **Feedback codes are returned by most callable services when requested.**
  - Indicate success or failure of service
  - Feedback codes may be omitted when calling a service
    - Language Environment will terminate if result of service is a severity 2 or greater feedback code.

Overview...

- ## Call syntax COBOL
  - CALL "CEExxxxx" USING parm1, parm2, ...,fc
  - ### The feedback code may be omitted
    - CALL "CEExxxxx" USING parm1, parm2, ...,OMITTED
  - ### Include file
    - COPY CEEIGZCT

Overview...

- ## Call syntax PL/I
  - Call CEExxxxx(parm1, parm2, ...,fc)
  - ### The feedback code may be omitted
    - Call CEExxxxx(parm1, parm2, ..., *)
  - ### Include file
    - %INCLUDE CEEIBMAW
    - %INCLUDE CEEIBMCT

Overview...

- **Call syntax C/C++**
  - CEExxxxx(&parm1,&parm2, ..., &fc)
  - The feedback code may be omitted
    - CEExxxxx(&parm1,&parm2, ...,NULL)
  - Include file
    - #include <leawi.h>
    - #include <ceeedcct.h>

Overview...

- **Call syntax Assembler**
  - CALL CEExxxxx
    - Parms are OS style with R1 pointing to parmlist
      LA 1,PARM1
      ST 1,PARMLIST
      LA 1,PARM2
      ST 1,PARMLIST+4

      ...
      LA 1,PARMLIST
  - The feedback code may be omitted
    LA 1,0
    ST 1,PARMLIST+xx
  - Include file
    - CEEBALCT

Math functions

- ## Names are of the form

  - ### CEESxyyy

    - Where x is the precision

I	Integer 32 bit
S	Single 32 bit floating point
D	Double 64 bit floating point
Q	Extended 128 bit floating point

      - Complex types also available

Math functions…

- **Names are of the form**
  - CEESxyyy
    - Where yyy is the function name

ABS	Absolute value
COS	Cosine
EXP	Exponential base e
LG1	Log base 10
SIN	Sine
SQT	Square root

      - Many many more

Math Functions...

- **Example CEESSSQT (Square Root)**

```
WORKING-STORAGE SECTION.
 01  FC.
    02  CONDITION-TOKEN-VALUE.
    COPY  CEEIGZCT.
        03  CASE-1-CONDITION-ID.
            04  SEVERITY      PIC S9(4) BINARY.
            04  MSG-NO        PIC S9(4) BINARY.
        03  CASE-SEV-CTL    PIC X.
        03  FACILITY-ID     PIC XXX.
    02  I-S-INFO               PIC S9(9) BINARY.
 01 MY-INPUT                   COMP-1.
 01 RESULT                     COMP-1.
 01 RESULT2                    PIC ZZZZ.99.
```

Math Functions...

- Example CEESSSQT (Square Root)...

```
PROCEDURE DIVISION.
    MAIN-PROG.
            DISPLAY "IN MAIN PROG".
            MOVE 9 TO MY-INPUT.
            CALL "CEESSSQT" USING MY-INPUT, FC, RESULT.
            IF NOT CEE000 OF FC THEN
                DISPLAY "SQUARE ROOT FAILED."
            END-IF.
            MOVE RESULT TO RESULT2.
            DISPLAY "THE SQUARE ROOT OF 9 IS:    " RESULT2.
```

Math Functions...

- ## Example CEESSSQT (Square Root)...

```
…
MOVE 144 TO MY-INPUT.
CALL "CEESSSQT" USING MY-INPUT, FC, RESULT.
IF NOT CEE000 OF FC THEN
   DISPLAY "SQUARE ROOT FAILED."
END-IF.
MOVE RESULT TO RESULT2.
DISPLAY "THE SQUARE ROOT OF 144 IS:  " RESULT2.
…
```

Math Functions...

- ## Example CEESSSQT (Square Root)...

```
...
MOVE -99 TO MY-INPUT.
CALL "CEESSSQT" USING MY-INPUT, FC, RESULT.
IF NOT CEE000 OF FC THEN
   DISPLAY "SQUARE ROOT FAILED."
END-IF.
MOVE RESULT TO RESULT2.
DISPLAY "THE SQUARE ROOT OF -99 IS:  " RESULT2.
GOBACK.
```

Math Functions...

- ## Example CEESSSQT (Square Root)... Output.

```
IN MAIN PROG
THE SQUARE ROOT OF 9 IS:        3.00
THE SQUARE ROOT OF 144 IS:     12.00
THE SQUARE ROOT OF 2500 IS:    50.00
SQUARE ROOT FAILED.
THE SQUARE ROOT OF -99 IS:     50.00
```

Outputting messages

- Several services are available to output messages.
  - CEEMGET
    - Get a message for a condition token.
  - CEEMOUT
    - Output a message from a string
  - CEEMSG
    - Output a message for a condition token

Outputting messages…

- We can use CEEMSG to output a Language Environment error message associated with the failure.

```
…
MOVE -99 TO MY-INPUT.
CALL "CEESSSQT" USING MY-INPUT, FC, RESULT.
IF NOT CEE000 OF FC THEN
  CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
  STOP RUN
END-IF.
MOVE RESULT TO RESULT2.
DISPLAY "THE SQUARE ROOT OF -99 IS:  " RESULT2.
…
```

Outputting messages...

- The output now looks like:

```
IN MAIN PROG
THE SQUARE ROOT OF 9 IS:      3.00
THE SQUARE ROOT OF 144 IS:    12.00
THE SQUARE ROOT OF 2500 IS:   50.00
CEE2010E The argument was less than 0 in
math routine SQRT    .
```

Outputting messages...

- We can use CEEMOUT to output a string.
- String must be a 2 byte prefix string

```
…
01 MY-MSG-STRING.
  02  MY-STRING-LEN           PIC S9(4) BINARY.
  02  MY-STRING-TEXT.
    03  MY-STRING-CHAR     PIC X,
                  OCCURS 0 TO 256 TIMES
                  DEPENDING ON MY-STRING-LEN
                     OF MY-MSG-STRING.
…
```

Outputting messages...

- **Must move length first.**

```
…
PROCEDURE DIVISION.
MAIN-PROG.
   MOVE 16 TO MY-STRING-LEN
           OF MY-MSG-STRING.
   MOVE "IN MAIN PROGRAM!" TO MY-STRING-TEXT
                       OF MY-MSG-STRING.
   CALL "CEEMOUT" USING MY-MSG-STRING, MSG-DEST, FC.
   IF NOT CEE000 OF FC THEN
     CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
     STOP RUN
   END-IF.
 …
```

Collecting error information...

- Output looks like.

```
IN MAIN PROGRAM!
THE SQUARE ROOT OF     9.00 IS:      3.00
THE SQUARE ROOT OF  144.00 IS:     12.00
THE SQUARE ROOT OF 2500.00 IS:   50.00
CEE2010E The argument was less than 0 in
    math routine SQRT    .
```

Collecting error information

- ## CEE3DMP service.
  - ### Requests a CEEDUMP be taken
    - Provide your own dump title
    - Pass options that control which information is included in the dump
      - Options we use when TERMTHDACT(TRACE)
        - NOENTRY COND TRACE THR(ALL) NOBLOCK NOSTOR TRCE GENOPTS
      - Options we use when TERMTHDACT(DUMP)
        - NOENTRY COND TRACE THR(ALL) BLOCKS STOR GENOPTS

Collecting error information...

- **CEE3DMP service.**
  - Recommended options when you call CEE3DMP.
    - ENCLAVE(ALL)   default
    - THREAD(CURRENT)   default
    - TRACE   default
    - FILES   default
    - VARIABLES   default
    - NOBLOCKS   default
    - NOSTORAGE   default
    - NOCONDITION   not default
    - ENTRY   default
    - GENOPTS   default

Collecting error information...

- ## CEE3DMP service.

  - ### Add service to our program.

```
…
01 DUMP-TITLE                      PIC X(80).
01 DUMP-OPTIONS                    PIC X(255).
PROCEDURE DIVISION.
MAIN-PROG.
  MOVE "SAMPLE DUMP TAKEN BY CEE3DMP." TO DUMP-TITLE.
  MOVE "NOCOND" TO DUMP-OPTIONS.

...
```

Collecting error information...

- CEE3DMP service.
  - Add service to our program.

```
…
MOVE -99 TO MY-INPUT.
CALL "CEESSSQT" USING MY-INPUT, FC, RESULT.
IF NOT CEE000 OF FC THEN
   CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
   CALL "CEE3DMP" USING DUMP-TITLE, DUMP-OPTIONS, FC
   STOP RUN
END-IF.
…
```

Collecting error information...

- Output looks like.

```
IN MAIN PROGRAM!
THE SQUARE ROOT OF     9.00 IS:      3.00
THE SQUARE ROOT OF  144.00 IS:     12.00
THE SQUARE ROOT OF 2500.00 IS:    50.00
CEE2010E The argument was less than 0 in
   math routine SQRT     .
```

Collecting error information...

- ## We also have CEEDUMP output

```
COMMAND INPUT ===>
PREFIX=*   DEST=(ALL)   OWNER=JMONTI   SYSNAME=
NP     DDNAME     StepName ProcStep DSID Owner
       JESMSGLG JES2                       2 JMONTI
       JESJCL     JES2                     3 JMONTI
       JESYSMSG JES2                       4 JMONTI
       SYSPRINT STEP1     COBOL       101 JMONTI
       SYSPRINT STEP1     LKED        102 JMONTI
       CEEDUMP  STEP1     GO          104 JMONTI
       SYSOUT    STEP1     GO          106 JMONTI
```

Date and Time information

- Language Environment provides a robust assortment of date and time services.

  - All services based on Lilian time – seconds/days since October 14, 1582
  - Valid through December 31, 9999
  - This timeframe chosen since it is a superset of all supported language time ranges.

Date and Time information…

- Let's add some to our program.

```
…
01 MY-DATE-LILIAN              PIC S9(9) BINARY.
01 MY-SECS-LILIAN             COMP-2.
01 MY-TIME-GREGORIAN         PIC X(17).
01 MY-PIC-STRING.
   02  MY-STRING-LEN          PIC S9(4) BINARY.
   02  MY-STRING-TEXT.
      03  MY-STRING-CHAR     PIC X,
                   OCCURS 0 TO 256 TIMES
                   DEPENDING ON MY-STRING-LEN
                       OF MY-PIC-STRING.
01 MY-TIMESTAMP              PIC X(80).
```

Date and Time information...

- ## Let's call CEELOCT to get the current local time.

```
…
CALL "CEELOCT" USING MY-DATE-LILIAN, MY-SECS-LILIAN,
                     MY-TIME-GREGORIAN, FC.
IF NOT CEE000 OF FC THEN
    CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
    CALL "CEE3DMP" USING DUMP-TITLE, DUMP-OPTIONS, FC
    STOP RUN
END-IF.
```

Date and Time information...

- Now convert MY-SECS-LILIAN to characters with CEEDATM.

```
...
MOVE 22 TO MY-STRING-LEN OF MY-PIC-STRING.
MOVE "MM/DD/YYYY HH:MI:SS AP" TO MY-STRING-TEXT
                                OF MY-PIC-STRING.
CALL "CEEDATM" USING MY-SECS-LILIAN,
                     MY-PIC-STRING,
                     MY-TIMESTAMP, FC.
IF NOT CEE000 OF FC THEN
   CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
   CALL "CEE3DMP" USING DUMP-TITLE, DUMP-OPTIONS, FC
   STOP RUN
END-IF.
```

Date and Time information...

- **Now output MY-TIMESTAMP with CEEMOUT.**

```
…
MOVE 22 TO MY-STRING-LEN OF MY-MSG-STRING.
MOVE MY-TIMESTAMP TO MY-STRING-TEXT
                      OF MY-MSG-STRING.
CALL "CEEMOUT" USING MY-MSG-STRING, MSG-DEST, FC.
IF NOT CEE000 OF FC THEN
   CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
   CALL "CEE3DMP" USING DUMP-TITLE, DUMP-OPTIONS, FC
   STOP RUN
END-IF.
```

Date and Time information...

- **Let's terminate with an ABEND**

```
…
MOVE 1234 TO ABDCODE
MOVE 0 TO TIMING

…
CALL "CEEMOUT" USING MY-MSG-STRING, MSG-DEST, FC.
IF NOT CEE000 OF FC THEN
    CALL "CEEMSG" USING FC, MSG-DEST, OMITTED
    CALL "CEE3DMP" USING DUMP-TITLE, DUMP-OPTIONS, FC
    CALL "CEE3ABD" USING ABDCODE , TIMING
END-IF.
```

Date and Time information…

- **Job output now shows U1234 ABEND:**

```
08.35.57 JOB05201  IEA995I SYMPTOM DUMP OUTPUT  360
   360              USER COMPLETION CODE=1234 REASON CODE=00000000
   360              TIME=08.35.57  SEQ=10636  CPU=0000  ASID=0037
   360              PSW AT TIME OF ERROR  078D1000  A0CC401A  ILC 2  INTC 0D
   360               ACTIVE LOAD MODULE        ADDRESS=20C29260  OFFSET=0009ADBA
   360               NAME=CEEPLPKA
   360               DATA AT PSW  20CC4014 - 00181610  0A0D58D0  D00498EC
   360               AR/GR 0: 9856AFDE/84000000   1: 00000000/840004D2
   360                 2: 00000000/A0EA94D8   3: 00000000/20EA94D0
   360                 4: 00000000/20C265B8   5: 00000000/20C26140
   360                 6: 00000000/20E86BE8   7: 00000000/20E890E0
   360                 8: 00000000/00000002   9: 00000000/20E85448
   360                 A: 00000000/20EA94D0   B: 00000000/A0CC3EF8
   360                 C: 00000000/20C27908   D: 00000000/20E893C0
   360                 E: 00000000/A0CD9884   F: 00000002/00000000
   360              END OF SYMPTOM DUMP
08.35.57 JOB05201  IEF450I JMONTIGO GO - ABEND=S000 U1234 REASON=00000000  361
```

Date and Time information...

- Our final output looks like the following along with our CEEDUMP from the CEE3DMP call:

```
02/14/2003 03:36:04 PM

IN MAIN PROGRAM!

THE SQUARE ROOT OF    9.00 IS:     3.00

THE SQUARE ROOT OF  144.00 IS:    12.00

THE SQUARE ROOT OF 2500.00 IS:    50.00

CEE2010E The argument was less than 0 in math
   routine SQRT    .
```

Other services

- **Many other services are available.**
  - Condition handling services
  - National Language services
  - Storage services
  - And more…

Summary

- Details on callable services available in the Language Environment Programming Reference. (SA22-7562-03)

- Callable services available from language environment conforming compilers

- CEE* available on any platform

- CEE3* available only on z/OS, z/VM

- Feedback codes report results of service

Appendix

- Full COBOL example
- Full PL/I example
- Full C example
- Full Assembler example