







Optimising
COBOL for VSEⁿ applications
using
LE/VSEⁿ, Debug Tool for VSEⁿ
and
Providing dump data to 21CS

Laura Grinham
VSEⁿ Technical Support Senior Software Engineer

A circular logo with a yellow border, containing a brown animal (possibly a dog or cat) sitting on a blue globe, with palm trees on either side and a yellow banner at the bottom with the word "WORKSHOP" in red.

Agenda

-  Brief Look at LE/VSEⁿ and Run-Time Options
-  LE/VSEⁿ Callable Services
-  Generation of COBOL Compilation Sidefiles
-  Using Debug Tool for VSEⁿ to Analyse Application Code
-  Providing Dump Data to 21CS for Problem Resolution
-  Using Trace Facilities



Brief Look at LE/VSEⁿ and Run-Time Options

LE/VSEⁿ:

Provides a single environment for Applications

Common functions and callable services

High Level Languages – C, COBOL, PL/I & HLASM

Calling/nesting modules may be written in different language

Manages application initiation/termination:

- 1 instance for each enclave, storage, linkage, amode swapping managed by LE/VSEⁿ

Customised environment with use of run-time options

To determine which Run-Time options in effect during execution:

- Individual programs - add RPTOPTS(ON) run-time parameter on EXEC JCL

- Batch - enter D CEE,CEEDOPT on the VSEⁿ console

- OLTP - enter D CEE,CEECOPT on the VSEⁿ console, transaction ROPC or CLER

To clear/update Batch run-time options: S CEE,OPTS=option1,option2,... or S CEE,CLEAR

To clear/update OLTP, use transaction CLER



LE/VSEⁿ Callable Services



LE/VSEⁿ installation library contains routines that provide run-time callable services Used by COBOL, PL/1 and C. Also, Assembler routines can call these services with the CEEENTRY macro and subsequently assembled with HLASM

CEEⁿⁿⁿ – common across all LE-enabled platforms

CEE5ⁿⁿ – specific to VSEⁿ

Common, consistent definitions are provided by

Header files (#include ⁿⁿⁿ for C)

Copy files (COPY ⁿⁿⁿ for COBOL) &

Include files (%include ⁿⁿⁿ for PL/1)

e.g. CEEIGZCI - COBOL declarations for condition information block

CEEIGZCT - COBOL declarations for LE/VSEⁿ symbolic feedback codes

COBOL applications call these services - CALL "CEE^{xxxx}" USING parm1 , parm2 , ... , fb code

Sample code using these services provided in PRD2.SCEEBASE for each language



LE/VSEⁿ Callable Services (cont)



Advantages of using LE/VSEⁿ callable services:

not 'reinventing the wheel' – function code written and ready for use

consistency inside an application suite – definition/precision of variables

removal of duplicate code – functions may be required several times in each program

confidence in results – no potential errors caused by different results produced from code written by different people

Some examples (with their sample names in PRD2.SCEEBASE):

IGZT5ABD.C – CEE5ABD, force an abend

```
CALL 'CEE5ABD' USING <<abendcode >>, << 0 | 1 >>
```

0 = produce system dump or OLTP transaction dump, 1 = terminate LE, produce LE dump

IGZT5DMP.C – CEE5SMP, generate a dump

```
CALL 'CEE5DMP' USING <<title >>, <<dump options>>, <<fb code>>
```

IGZTCRHP.C – CEECRHP & CEEDSHP, obtain/destroy heap storage

```
CALL 'CEECRHP' USING <<heapid>>, <<heap size>>, <<increment size>>, <<option>>, <<fb code>>
```



LE/VSEⁿ Callable Services (cont)



Services to provide time values, used for accurate time critical processing

IGZTCBLD.C – CEECBLDY, convert Date to COBOL Integer Format

```
CALL 'CEECBLDY' USING <<char date>>, <<picture string>>, <<integer>>, <<fb code>>
```

IGZTISEC.C – CEEISEC, convert integers to seconds

```
CALL 'CEEISEC' USING <<year>>, <<month>>, <<days>>, <<hours>>,
<<minutes>>, <<seconds>>, <<millisec>>, <<outsecs>>, <<fb code>>
```

IGZTLOCT – CEELOCT, get current local date and time

```
CALL 'CEELOCT' USING <<lilian>>, <<secs>>, <<gregorian>>, <<fb code>>
```

See also IGZTDATE, IGZTDATM, IGZTDAYS, IGZTDYWK



LE/VSEⁿ Callable Services (cont)



For multi-national companies, applications can be written to run seamlessly in more than one country using LE/VSEⁿ's National Language Support (NLS) and Locale Callable Services.

NLS Support, similar to Run-Time options COUNTRY and NATLANG.

IGZT5CTY – CEE5CTY, query or set default national country settings for date & time format, currency symbol and decimal/thousands separator.

```
CALL 'CEE5CTY' USING <<function>>, <<country code>>, <<fb code>>
```

See also:

IGZT5LNG.C (CEE5LNG – query/set national language)

IGZT5MCS.C (CEE5MCS – query default currency symbol), select correct code page

IGZT5MDS.C (CEE5MDS – query default decimal separator)

IGZT5MTS.C (CEE5MTS – query default thousands separator)

IGZTFMDA.C (CEEFMDA – query default date format)

IGZTFMDT.C (CEEFMDT – query default date and time format)

IGZTFMTM.C (CEEFMTM – query default time format)



LE/VSEⁿ Callable Services (cont)



Locale Callable Services – provide output appropriate for each country
These functions equivalent to C functions, e.g.

IGZTFMON.C <-> strfmon()	formats monetary string
IGZTFTDS.C <-> strftime()	formats date/time string
IGZTLCNV.C <-> localeconv()	query locale numeric attributes
IGZTQDTC.C <-> localdtconv()	query locale date/time attributes
IGZTQRYL.C <-> setlocale()	query active locale environment
IGZTSETL.C <-> setlocale()	sets locale operating environment
IGZTSCOL.C <-> strcoll()	compares collation sequence
IGZTSTXF.C <-> strxfrm()	translates string characters to collation sequence

Services to provide access to parameters

IGZT5PRM – CEE5PRM, query parameter string passed to program, see also IGZTPRL
CALL 'CEE5PRM' USING <<parm string>>, <<fb code>>

IGZT5USR – CEE5USR, set or query user fields
CALL 'CEE5USR' USING <<function>>, <<field #>>, <<value>>, <<fb code>>



LE/VSEⁿ Callable Services (cont)



Call to initiate debugging tool e.g. Debug Tool for VSEⁿ - CEETEST

Call to use HDB for VSEⁿ - CEETDLI , similar to using the CBLTDLI interface

Services for bit manipulation

CEESICLR (clear), CEESISSET (set), CEESISHF (shift) & CEESITST (test)

(disclaimer.... Never tried it!)

Use of available Maths functions would ensure consistency without unintended errors due to rounding, loss of significant digits, etc:

Examples of Maths functions:

Absolute Value, Sine & Cosine, Floating Point Complex Divide & Floating Point Complex Multiply, Truncation, Exponentiation, Exponential Base, Logarithmic Base, Square Root



LE/VSEⁿ Callable Services (cont)



There are implications for linking

Applications are generated with AMODE/RMODE attributes:

AMODE - the addressing mode (24-bit, 31-bit or any bit-addressing)

RMODE - the residence mode (below 16Mb, any location above/below 16Mb)

Associated Run-Time option:

ALL31: ON - application runs in 31-bit addressing,
 OFF - some part must run in 24-bit addressing

LE/VSEⁿ's callable services run in 31-bit addressing mode

Automatic mode switching occurs when an application running in 24-bit addressing requests an LE callable service and **ALL31** is set to OFF.

To avoid automatic mode switching and therefore improve performance re-link the program to run in 31-bit addressing mode and set **ALL31** to ON

See Appendix A of LE Programming Guide for guidance to write own Callable Services



Generation of COBOL Compilation Sidefiles



COBOL compilation listings, by default are printed on syslst.

Compile time TEST option allows compilation to generate a sidefile.

Sidefile saved in VSEⁿ library with filetype of 'SYSDEBUG'

e.g. // EXEC IGYCRCTL,SIZE=IGYCRCTL,PARM='SD(USERLIB.TESTIVP(LGSEP))'
 CBL QUOTE TEST(NONE,SYM,SEPARATE) NAME

Debug Tool for VSEⁿ uses the listing or source.

For C, source is cataloged into VSEⁿ library

For COBOL and PL/1, the compilation listing is used.

Debug Tool for VSEⁿ does not support the use of COBOL sidefiles.

Debugging COBOL programs compiled with the SEPARATE sub-option of the TEST compile time option generates an error.

Compile with the NOSEP test sub-option for debugging purposes and if required, recompile with the SEP test sub-option before migrating to production.



Using Debug Tool for VSEⁿ to Analyse Application Code



LE/VSEn Run-Time option

TEST(<< condition level >> , << commands >> , << prompt level >>, << preference >>)

condition level to trigger when control passed to a debugging tool – all / error / none

commands/file name – initial debugging commands to run when triggered

prompt – invoke debug tool at LE initiation before program execution

session preferences/file name – commands to establish session environment

Commands and Session Preferences may be on sysipt or in a previously created file

Not to be confused with the TEST **Compiler** Option for each of the HLLs which specify locations for hooks/breakpoints and whether the Symbol Table is generated

COBOL TEST(<<hook-location>>,<<sym>>,<<sysdebug>>)

hook-location – one of block, path, statement, all, none

sym – create symbol table

sysdebug (discussed above)



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Listings are made available to Debug Tool for VSEⁿ (DT) via a printer exit at compile time

```
// EXEC IGYCRCTL,SIZE=IGYCRCTL,PARM='EXIT(PRTEXTIT(EQUALIST))'
```

The printer exit generates a library member or a SAM ESDS file in VSAM managed space

By default, EQUALIST creates a sub-library member <<cuname>>.list

Run-time TEST options not specified as parameters on JCL EXEC card for OLTP, HDB or SDB

Use the run-time default modules – CEEDOPT (batch), CEECOPT (OLTP) or CEEUOPT (user)

DT can be executed in 4 modes:

- Batch

- Batch Interactive

- OLTP – single terminal (debugging session on same terminal as DT)

- OLTP – dual terminal (debugging session on a different terminal as DT)



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Batch – runs to completion in partition, all DT commands and output in job output.

```
// EXEC IVPJOB1,PARM=/' TEST(ALL,SYSIPT,PROMPT,SYSIPT)'  
    * Session Preference Input  
    SET MSGID ON;  
/*  
    * Command file Input  
    AT 98  IF R = 7 THEN LIST TITLED (R, VARBL1, VARBL2);  END-IF;  
    AT 101 PERFORM;  
        LIST TITLED ( %LINE, "END OF LOOP", R );  
        LIST ("ABOUT TO QUIT", %CU );  
        QUIT;  
    END-PERFORM;  
    AT EXIT * LIST ( "Exiting ", %CU );  
    LIST STR1;  
    DESCRIBE ATTRIBUTES STR1;  
    77 TEMP PIC X(5);  
    MOVE STR1 TO TEMP;  
    LIST TEMP;  
    CLEAR AT 88;  
/*
```

DT gains control at job initiation – prompt sub-option
Session environment set up by commands on sysipt
DT debug commands entered on sysipt



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Batch Interactive – job runs in partition, pauses when DT is given control, DT session on VCDD terminal

Provided source VCDD B.book EQAWAPPL.B defines 20 terminal IDs for use by DT

Catalog definitions as a minor node in VTAM ATCCONxx in PRD2.CONFIG

Restart VCDD or VARY NET,ACT,ID=EQAWAPPL

```
// EXEC IVPJOB2,SIZE=AUTO,PARM='/TEST(ALL,SYSIPT,;,MFI%D0810001:*)'
```

```
* session environment setting (preference file)
```

```
SET PACE 1;
```

```
SET MSGID ON;
```

```
LIST %NLANGUAGE;
```

```
/*
```

```
* Command file Input
```

```
MONITOR LIST (VARBL1, VARBL1, STR1, STR2, STR3, R);
```

```
MONITOR LIST REGISTERS;
```

```
STEP 6;
```

VCDD terminal has to be attached to VSEⁿ (dialled as cuu **081**)

DT gains control after session environments commands entered on sysipt, have completed

Control passed to debugging session on VCDD terminal



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



OLTP

Update CSD with definitions for DT programs/transactions provided in EQACCSD.z, in group EQA
Restart OLTP or install group EQA using transaction CEDA

To use the provided OLTP IVPs, update CSD with required program and transaction, e.g.

```
CEDA DEFINE PROG(IVPCOB3) LANG(COBOL)   GROUP(EQAIVP)
CEDA DEFINE TRANS(IVPC)   PROG(IVPCOB3)  GROUP(EQAIVP)
```

Restart OLTP or install group EQAIVP using transaction CEDA

To debug in Dual Terminal mode, the provided user exit EQADCCXT is linked into the OLTP program.

Run transaction DTCN, enter terminal IDs for user transaction and DT session

Optionally enter test run-time options

Start user transaction

Transaction will animate on the specified terminal ID and DT session will have control on the other.



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Initial display of OLTP DTCN transaction screen

Enter required values, PF4 – Add, creates the profile for this debugging session

DTCN profile saved in working storage (not retained after OLTP restart)

```
DTCN                DEBUG TOOL OLTP Interactive Runtime Facility                PRODCICS

Item                Choice                Possible choices

Terminal Id         ==> A000                Application Terminal Id
Transaction Id      ==>                    Any valid Trans Id

Session Parm
DT VSEn Term Id    ==>                    MFI - DT Term Id (dual terminal mode)

Test Option         ==> Test                Test/Notest
Test Level          ==> All                All/Error/None
Command File        ==>
Prompt Level        ==> Prompt
Preference File     ==> *

Any other valid Language Environment Options
==>

EQA2007E Show failed - profile does not exist

PF1=HELP 2=GHELP 3=EXIT 4=ADD 5=REPLACE 6=DELETE 7=SHOW 8=NEXT 10=CLOSE DTCN
```



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



- AT <<line#>> Set breakpoint at point in logic which is suspect.
- AT <<line#>> IF <<variable name>> > -1 THEN GO ; END-IF
 Set conditional breakpoint – error only occurs for certain value/conditions
- MONITOR REGISTERS Track addressing, etc.
- 77 TEMP PIC X(5); Define temporary variables during debugging session
 perform calculations or moves etc. with temporary variables
- COMPUTE <<tempvar>> = (var1 + array(1)) / c * 2
 results of COBOL statements available in real time (some limitations)
- AT CHANGE %STORAGE(H'00521D42',8)
 Check for storage overlays, either at absolute address or using variable name
- LIST CALLS Trace back through program logic flow
- LIST %ADDRESS List address at point of interruption
- At * LIST %LINE track of all lines executed - use with discretion



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Initial, default display of debugging session - quite pretty... or maybe 'pretty garish'!

The colours, size and location of each window can be modified, e.g. SET COLOR

Session environment commands entered on sysipt or in a 'preference' file <<user>>.DTPREF

```
COBOL LOCATION: IVPCOB2 :> 89.1
Command ==>
MONITOR --+---1---+---2---+---3---+---4---+---5---+---6---+---7---
0001 1 VARBL1 10
0002 VARBL1 10
0003 Invalid data for 77 IVPCOB2:>STR1 is found.
0004 Invalid data for 77 IVPCOB2:>STR2 is found.
0005 Invalid data for 77 IVPCOB2:>STR3 is found.
0006 R 1
0007 2 GPR 0 - 3 = 00469120 00000000 004690D0 00000000
0008 GPR 4 - 7 = 004000B0 00469018 0041E778 00000000
SOURCE: IVPCOB2 --1---+---2---+---3---+---4---+---5---+---6---+---7---
89 MOVE "ONE" TO STR1. MOVE "TWO" TO STR2.
90 MOVE "THREE" TO STR3.
91 MOVE "FOUR" TO STR4. MOVE "FIVE" TO STR5.
92 PERFORM UNTIL R = 9
93 MOVE "TOP" TO STR1
94 MOVE "BEG" TO STR2
95 MOVE "UP" TO STR3
LOG 0---+---1---+---2---+---3---+---4---+---5---+---6---+---7---
0024 MONITOR
0025 LIST REGISTERS ;
0026 GO ;
0027 AT THE BREAKPOINT FOR LINE
0028 %LINE = 89.1
PF 1:? 2:STEP 3:QUIT 4:LIST 5:FIND 6:AT/CLEAR
PF 7:UP 8:DOWN 9:GO 10:ZOOM 11:ZOOM LOG 12:RETRIEVE
```

An example of listing raw storage:

```
LOG 0---+---1---+---2---+---3---+---4---+---5---+---6---+---7---
0027 AT THE BREAKPOINT FOR LINE
0028 %LINE = 89.1
0029 LIST STORAGE ( H'00469120', 32 ) ;
0030 00469120 00000000 004691F4 0043444F 00531010 *.....j4...|....*
0031 00469130 004691A0 00000004 00401C08 00401C08 *..j..... ..*
```



Using Debug Tool for VSEⁿ to Analyse Application Code (cont)



Common, often used commands can be saved in a commands file.

Passed to DT as a sub-option of the TEST run-time option, e.g.

```
// EXEC .... ,PARM='/TEST(ALL,"USERLIB.SUBLIB(DTFILE.CMDS)",PROMPT,*)
```

or with USE command, e.g.

```
USE USERLIB.SUBLIB(DTFILE.CMDS) ;
```

Session log can be saved for all debugging modes, e.g. SET LOG ON FILE <<file name>>;

Log file subsequently input as commands file to enable repeatable, consistent debugging

Session customisations (profiles) for each user are automatically saved and updated

e.g. PF key assignments, display colours.

Profiles saved in library in Source search chain as <<user>>.DTSAFE



Providing Dump Data to 21CS for Problem Resolution



System Dumps etc, in VSEⁿ Dump Management System – IUI, fast path 4.3

naming convention: SYSDUMP.<<pn>>.<<dump #>>

Dumps can be offloaded, deleted, or printed.

In the first instance, provide a full dump in binary format, LRECL of 4112.

Save it in your BOX folder: From <<customer name>> to 21CS debug data upload

That allows us to run dump analysis against the raw data:

VM utility – VSEDUMP

VSE Tool – INFOANA



Providing Dump Data to 21CS for Problem Resolution (cont)



We may ask for printed dumps generated by VSE Dump Management (IUI), e.g.,

```
// JOB DMPSYM2 PRINT DUMP SYMPTOMS
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=300K
  SELECT DUMP MANAGEMENT
    DUMP NAME SYSDUMP.BG.DBG00021
    RETURN
  SELECT DUMP SYMPTOMS
    PRINT DATA
    RETURN
  SELECT END
/*
```

Or specific domain/kernel area, e.g., KE = OLTP kernel, AP=Application Domain (DEF default)

Trace level 0=suppress, 1=summary, 2=full, 3=summary and full

```
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=1024K
  DUMP NAME SYSDUMP.F3.DF300030
  SELECT DUMP VIEWING
    CALL DFHPD440 DATA DEF=0,KE=3, AP=1,DS=1,LD=1,TR=1
/*
/ &
```



Providing Dump Data to 21CS for Problem Resolution (cont)



IF... LE/VSEⁿ abends in OLTP => abend code 4000-4095

Enter CEEABENDC at command prompt on VSEⁿ console, PF9 Explanation key

```
SYSTEM: VSEn          VSEn 6.3          TURBO (01)          USER: SYSA
VM USER ID: PTHVSE37          TIME: 12:24:46
CEEABENDC

The following LE/VSE Abend Codes are available via EXPLAIN-key.

The four-digit decimal abend code is specified after the "CEEU..."prefix.
The three-digit hexadecimal abend code is shown in parentheses.

Move cursor to the required CEEUxxx and press the EXPLAIN-key to see the
explanation for the abend code.

CEEU1xxx

CEEUxxxx

CEEU4000 (X'FA0')

CEEU4034 (X'FC2')
```



Providing Dump Data to ZICS for Problem Resolution (cont)



Scroll to relevant abend code, PF9 Explanation key displays following:

```
SYSTEM: VSEn          VSEn 6.3          TURBO (01)          USER: SYSA
VM USER ID: PTHVSE37          TIME: 12:26:44
CEEU4042 (X'FCA')

Explanation: The application heap data has been corrupted.

Reason codes:

0 (X'00')
  The LE for VSEn heap checker has detected that heap storage has
  been corrupted.

1 (X'01')
  An unrecoverable error occurred while the heap checker was
  attempting to check for, or report on, heap damage.

System Action: Enclave terminated with a dump.
```



Providing Dump Data to 21CS for Problem Resolution (cont)



In addition to the sample jobs in LE/VSEⁿ library PRD2.SCEEBASE which generate the default LE Run-Time Options modules:

CEEWDOPT.z – installation wide defaults for BATCH programs

CEEWCOPT.z – installation wide defaults for OLTP programs

CEEWUOPT.z – application specific defaults, assembled then linked into phase

CEEXUOPT.z generates a dump when LE/VSEⁿ abends in OLTP for abend codes listed by CEEABENDC (4000 – 4095)

...

CEEXOPT ABPERC=(Uxxxx),

ABTERMENC=(ABEND),

RPTSTG=(ON),

TERMTHDACT=(UADUMP,LSTQ,96),

TRAP=(ON,MAX)

...



Using Trace Facilities



CEETRACE

Uses CEL4TRCE phase instead of default CEEBINT

Installation involves saving CEEBINT.phase (renaming as default or PRD2.SAVE)

save copy of CEL4TRCE.phase (rename or PRD2.SAVE)

install CEETRACE routine as LE VSEⁿ installation wide HLL exit

```
rename CEL4TRCE.phase:CEEBINT.phase
```

To use updated trace options, enter AR command

```
S CEE,CEETRACE=RELOAD
```

To stop trace

```
// EXEC DTRIATTN,PARM='S CEE,CEETRACE=OFF'
```

To display status, enter AR command

```
D CEE,CEESTAT
```

When required, Level 3 will provide specific trace statements.



Using Trace Facilities (cont)



OLTP Trace Facility – transaction code CETR

OLTP Auxiliary Trace:

Two trace tables: DFHAUXT & DFHBUXT.

SAM datasets, restart OLTP after definition

If problem occurs during OLTP startup, modify SIT with the following:

SYSTR=ON

AUXTR=ON

AUXTRSW=NO

STNTR=1 to turn level 1 tracing on for all OLTP components

Or.. STNTRxx=1 where xx is a specific component abbreviation

When OLTP running, use Transaction CELR

Auxiliary Trace Status ==> Started

Auxiliary Switch Status ==> Next or NO

Master System Trace Flag ==> On

PF4 displays component list, select level of tracing required,

Start and stop Auxiliary trace as required with CELR transaction

Process the Auxiliary Trace dataset using trace utility print program (DFHTU410) using same format as INFOANA (p22)



Using Trace Facilities (cont)



Initial display of OLTP CETR transaction screen

Overtyping values, enter to update

PF4 lists for selection, OLTP domain/kernel, etc at specified trace levels (see p 22)

PF5 allows selection of transactions/terminals for tracing

```

CETR                                CICS Trace Control Facility                                CIC2
Type in your choices.

Item                                Choice                                Possible choices
Internal Trace Status                ==>>> STARTED                        STArted, STOpped
Internal Trace Table Size            ==>>> 4096 K                            16K - 1048576K

Auxiliary Trace Status               ==>>> STOPPED                        STArted, STOpped, Paused
Auxiliary Trace Dataset              ==>>> A                                A, B
Auxiliary Switch Status              ==>>> NO                               NO, NExt, ALL

Master System Trace Flag             ==>>> ON                               ON, OFF
Master User Trace Flag               ==>>> ON                               ON, OFF

When finished, press ENTER.

PF1=Help      3=Quit      4=Components      5=Ter/Trn      9=Error List

```



Thank You

laura.grinham@21cs.com

support@21cs.com