SHARE February, 2008

Session Number	8241
Session Title	Migrating to COBOL compilers under LE
Speaker/Author	Tom Ross

Migration Key Points

I recommend that you have a 'COBOL DBA'

What is a run-time library?

Run-time migration vs source conversion

Product history/positioning

Run-time migration

COBOL source conversion

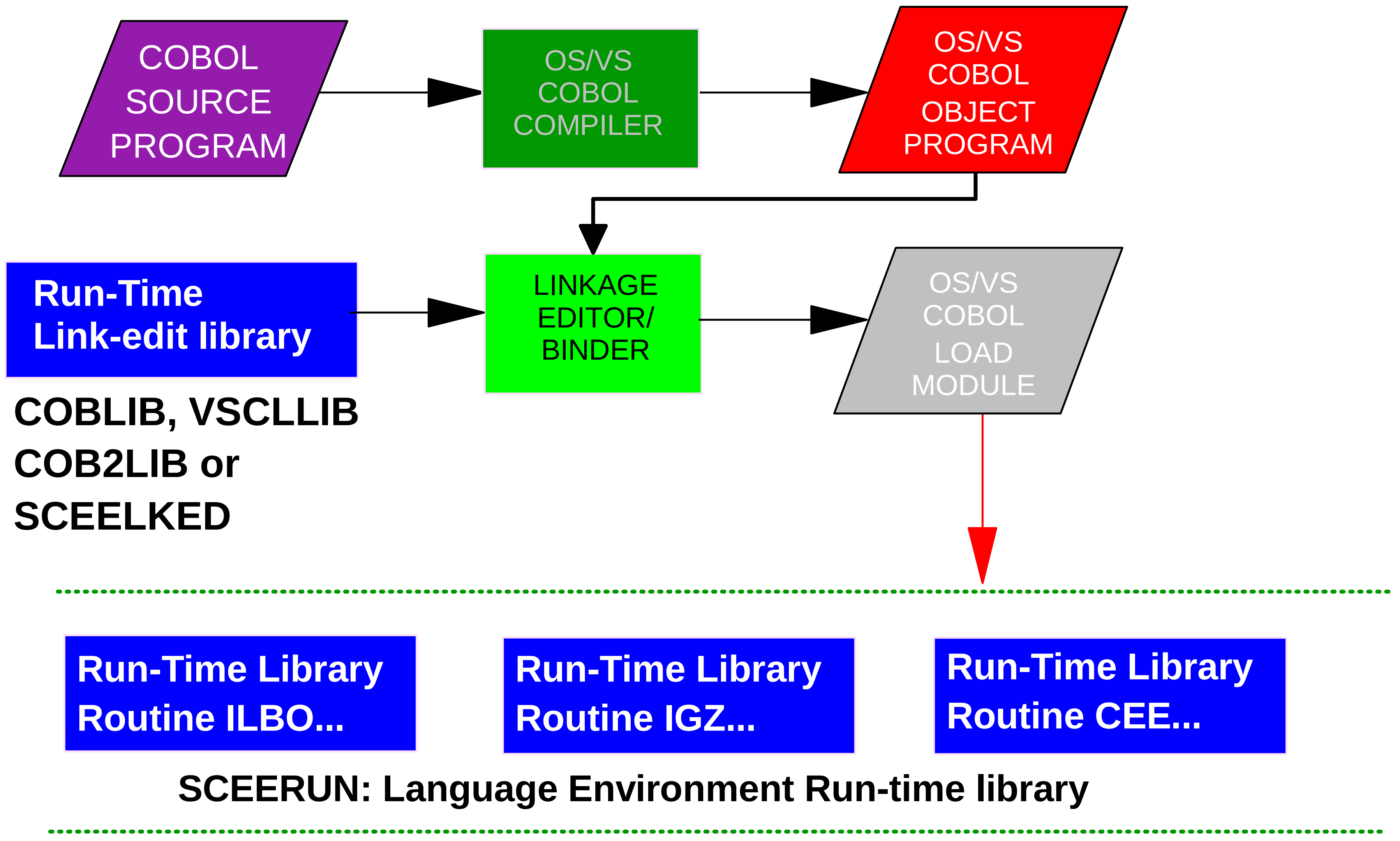
Benefits of migration

Technical advantages of LE

What is a language run-time library?

- **High-level languages require library routines in order to execute**
  - **Example: The code for file OPEN is not generated by the compiler, instead a call to a library routine is generated**
  - **Other Examples: INIT/TERM, COBOL variable length move, INSPECT, SORT, dynamic CALL, and more!**
- **For COBOL programs, the library routines have been largely invisible to application programmers and system programmers**
  - **The routines were either statically linked in to object modules (COBOL compiler option NORES)**
  - **Or the routines were accessed dynamically (COBOL compiler option RES) through LNKLST/LPALST, which meant that users did not have to write any JCL to get access to them**

What is a language run-time library?



COBOL SOURCE PROGRAM
OS/VS COBOL COMPILER
OS/VS COBOL OBJECT PROGRAM
Run-Time Link-edit library
LINKAGE EDITOR/ BINDER
OS/VS COBOL LOAD MODULE
COBLIB, VSCLLIB
COB2LIB or
SCEELKED
Run-Time Library Routine ILBO...
Run-Time Library Routine IGZ...
Run-Time Library Routine CEE...
SCEERUN: Language Environment Run-time library

Run-Time Migration

Service for OS/VS COBOL or OS PL/I V1 programs without service extension from IBM!

- **How?**
- **Every COBOL program requires library routines in order to execute**
  - **Example: ILBOVOC for COBOL VSAM OPEN/CLOSE**
- **Programs that are using supported versions of the library routines are supported by IBM service!**
  - **All OS/VS COBOL and VS COBOL II library routines are in LE**
- **Complete your migration without recompiling any programs!**

Programs supported by Language Environment

	Enterprise COBOL for z/OS Version **4** Programs			
	Enterprise COBOL for z/OS Version 3 Programs		Enterprise PL/I for z/OS Version 3 Programs	
	COBOL for OS/390 & VM Version 2 Programs	C/C++ MVS/ESA Programs	PL/I for MVS & VM (Multitask Programs)	
LE-conforming Assembler Programs	COBOL for MVS & VM Programs	AD/Cycle C/370 Programs	AD/Cycle PL/I MVS/VM Programs	
	VS COBOL II Programs R3 and R4	C/370 Ver. 2 Programs	OS PL/I Version 2 Programs (any rel.)	VS FORTRAN V1 and V2 Programs
Assembler Programs	OS/VS COBOL Programs (Rel 2.4)	C/370 Ver. 1 Programs (Rel. 2)	OS PL/I V1 R3 (object) R4 or later load or obj	OS FORTRAN G and H Programs
Assembler	COBOL	C	PL/I	FORTRAN

What is LE?
What is Enterprise COBOL ?

- LE has all OS/VS COBOL and VS COBOL II library routines
  - Packaged differently, same internal logic
  - Different options, storage usage, same results
  - Migrated modules either ABEND or get the right answer
  - No need to validate arithmetic results, etc.
- VS COBOL II uses the same routines as Enterprise COBOL
- Enterprise COBOL V4 is compiler only
  - Think of it as VS COBOL II Release 13
  - 99% compatible with VS COBOL II NOCMPR2
- and by the way, z/OS is just another MVS
- Don't worry!

What is LE? Apply maintenance correctly!

- IBM has been getting lots of calls caused by incorrect maintenance procedures at customer shops
  - In this case, compiler PTF UK13240 (SEARCH ALL)
  - Abend IGZ0099C
  - "Internal error 2 was detected in module IGZCLDR"

- The problem?
  - Customers install a compiler PTF that pre-reqs a COBOL run-time (LE) PTF, but only install the LE PTF in the development region, not in the test or production regions!

- The solution?
  - Make sure your installers know that LE is like DB2, IMS and CICS: if you fix it in one place, fix it everywhere!
  - When migrating to a new compiler, go through maintenance procedures to get PTFs on all systems FIRST, then use new compiler

What's this about SEARCH ALL?

- **LE APAR PQ95124 corrected errors in the COBOL SEARCH ALL statement**
  - **For all COBOL modules!**

- **However, users have found COBOL applications that depend on the erroneous behavior!**

- **New APARs:**
  - **PK19573/PK15432 (LE/COBOL library)**
  - **PK16765 (V3R4 compiler)**
    change the product so the corrected results are only generated for programs compiled with V3R4
    - Corrected results are built-in to Version 4

SEARCH ALL with prior COBOL releases:

1. If search argument longer than table key, in some cases argument matched key even when excess digits/characters were not zeros/blanks:
```
01 Arg1 pic X(8).      "ABCDxxxx".
  ...
  03 Key1 pic X(4).    "ABCD"
```
2. Signed search argument with negative value may match unsigned table key:
```
01 Arg2 pic S9(4).    -1234
   …
  03 Key2 pic 9(4).    1234
```
   Note: these should compare unequal! Just as in:
```
IF Key(IX) = Arg …
SEARCH … WHEN Key(IX) = Arg …
```
   (serial search, no ALL…)

   V3R4 compiled SEARCH ALL WHEN will compare unequal as well.

SEARCH ALL statement – migration

- V4 compiler (and V3R4 with PTFs for APAR PK16765) generates warning messages for SEARCH ALL statements that might have different results. Whether an actual change occurs depends on the contents of the search argument at run time
- LE/COBOL runtime (with PTFs for APAR PK15432) generates warning messages if search argument value can never match a table key
- Go to www.ibm.com/software/awdtools/cobol/zos/support/
  - Search for item 1243387

What about old COBOL and CICS TS?

- **CICS TS 2.2**
  - **Cannot translate OS/VS COBOL programs**
  - **OS/VS COBOL programs will run with old run-time libraries**
- **CICS TS 2.3**
  - **OS/VS COBOL programs will run, but only if using LE**
  - **Any attempts to initialize OS PL/I, OS/VS COBOL, or VS COBOL II run-times will result in ABEND**
  - **All HLL programs running under CICS must use LE**
- **CICS TS 3.1 and 3.2**
  - **OS/VS COBOL programs will abend even if using LE!**
  - **VS COBOL II not supported, but won't abend if using LE**
  - **Example: DFHAC2206 10:14:20 CICSC31F Transaction 1608 failed with abend ALIK.**
    - ➤ EXPLANATION: CICS has determined that an OS/VS COBOL program is about to be executed. However CICS no longer supports such programs.

What about old COBOL and DB2 V8/V9?

- **DB2 Version 8**
  - **Version 8 precompiler cannot be used with OS/VS COBOL**
  - **Only Enterprise COBOL V3R3 with PQ83744, V3R4 and V4R1 can use new SQL features with integrated coprocessor**
    - ➢ Use DB2 NEWFUN(YES) option
    - ➢ Example: multi-row FETCH or INSERT
    - ➢ Separate precompiler for DB2 V8 also supports new SQL functions
  - **Old applications will run unchanged once you do the database migration**
    - ➢ If you make program changes, use a supported compiler
  - **A precompiler that is very similar to the DB2 V7 precompiler is ported to DB2 V8 via PK46170.**
    - ➢ The V7 precompiler options processing is changed to issue an error for the restrictions in language and other options.
- **DB2 Version 9**
  - **Same story as V8 except must use V9 precompiler**
    - ➢ (either integrated or separate)

What about NEW COBOL and DB2 V8?

- **DB2 Version 8 Potential Performance Problem**
  - **Integrated coprocessor only (SQL compiler option)**
  - **You are required to tell DB2 what CODEPAGE CCSID your data is to be defined as**
    - ➤DSNHDECP
  - **Enterprise COBOL also requires you to tell us what your CODEPAGE CCSID is**
    - ➤CODEPAGE compiler option
  - **If these are NOT the same, then every column will be translated from one CCSID to the other on every data access. Example:**
    - ➤DSNHDECP says CCSID 037, COBOL uses CCSID 1140
    - ➤Even though there is only one character different (EURO), every column will be transalted using Unicode Conversion Services
  - **Conclusion: Do not use a different CCSID in COBOL with integrated coprocessor from your DB2 data!!!!!!!!!!!!!!**

What about old COBOL and Debug Tool?

- **Debug Tool Version 6 with Debug Tool Utilities and Advanced Functions provides COBOL Migration help**
  - **DTUAF Version 6: PID 5655-P15**
  - **Includes a Load Module Analyzer to find if you have OS/VS COBOL programs in load libraries**
  - **Includes COBOL Conversion Aid to convert 74 Std COBOL source code to 85 Std**
  - **Provides capability to debug any COBOL load modules when mixed together (for gradual migration to Enterprise COBOL)**
- **Debug Tool Version 7 has it too!**
  - **DTUAF Version 7: PID 5655-R45**

Migration Overview

- Everything you need to know is in COBOL Migration Guide
- Plan, plan, plan
- Assign COBOL 'DBA' (person or group): focal point
- Communicate with management and users
  - Make friends with your system programmer
- Migrate run time, then link edit library, then upgrade source
- Use old compiler and link edit library until new run time with pre-req PTFs for new compiler is in production
- Migrate in test environment first
- Regression test . . . why?
  - Your code
  - IBM code
  - Third party code

Migration Overview

Everything you need to know!

- **Missing COBOL source?**
  - **Source Recovery Inc.**
    - www.source-recovery.com/index.html
    - Jim Rahm (770) 667-5043
  - **HLASM toolkit disassembler**
- **Other resources**
  - **COBOL & LE IBM newsgroups**
    - See last page
  - **www.ibm.com**
    - /software/awdtools/cobol
    - /servers/eserver/zseries/zos/le/
- **Publications are all on the Web!**
  - **www.ibm.com/servers/eserver/zseries/zos/bkserv/**
- **Migration publications**
  - **IBM COBOL Compiler and Runtime Migration Guide (GC23-8527-00)**
  - **IBM PL/I Compiler and Run-Time Migration Guide (GC27-1458)**
  - **IBM C/C++ for MVS/ESA Compiler and Run-Time Migration (GC09-4913)**
  - **IBM Language Environment Run-Time Migration Guide (GA22-7565)**
  - **Call (800)879-2755 to order IBM publications**

Preparation for Migration

Take inventory of your applications

- **For each application, determine (by hand, or with a tool such as the IBM Load Module Analyzer that comes with Debug Tool, or EDGE Portfolio Analyzer):**
  - **level of source used**
    - ➤ CICS Macro Language
    - ➤ COBOL 68, 74, 85
  - **which compiler used, what release**
  - **compiler options and run-time options used**
  - **presence of inter-language communication (ILC)**
    - ➤ PL/I-COBOL and C-COBOL migration aids exist (see next page or migration guides)
  - **presence of assembler**

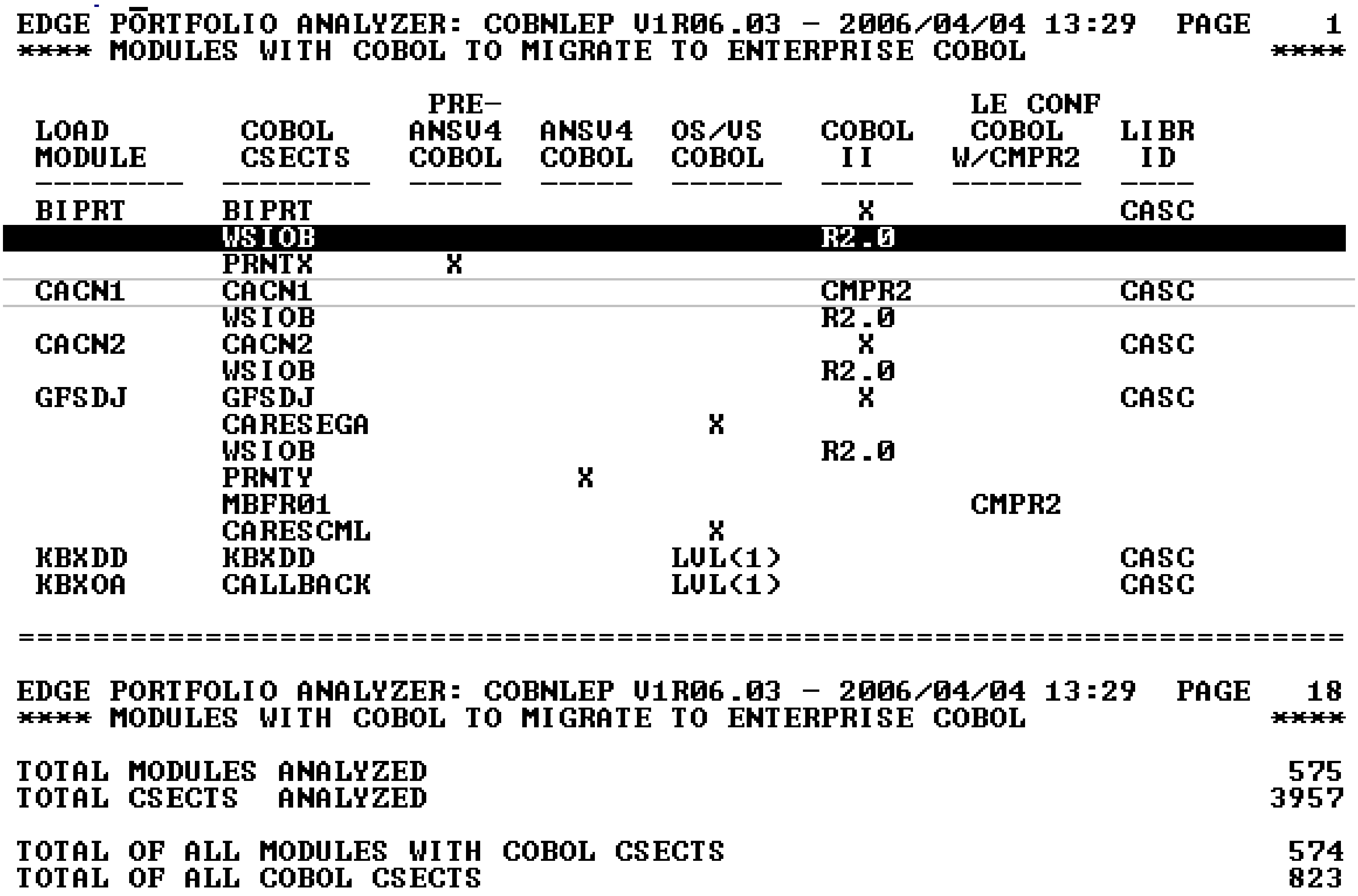
Preparation for Migration

- **IBM Load Module Analyzer (LMA) comes with IBM Debug Tool**
- **LMA Features:**
  - **Reports compiler versions and compile dates for all CSECTs that make up a load module**
  - **Can be executed interactively from ISPF panels or in batch**
  - **Report display options are configurable:**
    - ➤ Filter compiler versions reported
    - ➤ Filter CSECTs that are part of the LE runtime
- **Customers can get a “no charge, 30 day” evaluation copy of LMA to determine the amount of back level COBOL running in the production environment**
  - **Customers must sign an “Evaluation Agreement”**
  - **Contacts to acquire Evaluation Load Module Analyzer:**
    - ➤ Dan Brown, browndan@us.ibm.com
    - ➤ P.J. Baron, pbaron@us.ibm.com

Preparation for Migration

- **EDGE Portfolio Analyzer works with load modules**
  - **EDGE Portfolio Analyzer tool**
    - ➤ No longer available from IBM
    - ➤ Go to www.edge-information.com
  - **Scans load module libraries (any language)**
  - **Tells you what (if any) needs to be relinked**
  - **Build Linkage Editor control statements to aid relinking**
  - **Determines which compiler and what options were used**
- **Check maintenance level of current run-time library**
  - **PN74000 for VS COBOL II R4 bootstrap IGZEBST**
    - ➤ Mixing *VS COBOL II and newer COBOL*
  - **PN46223, PN69803 & PN69804 for OS PL/I V2**
    - ➤ For mixed PL/I and COBOL (ILC)

Preparation for Migration

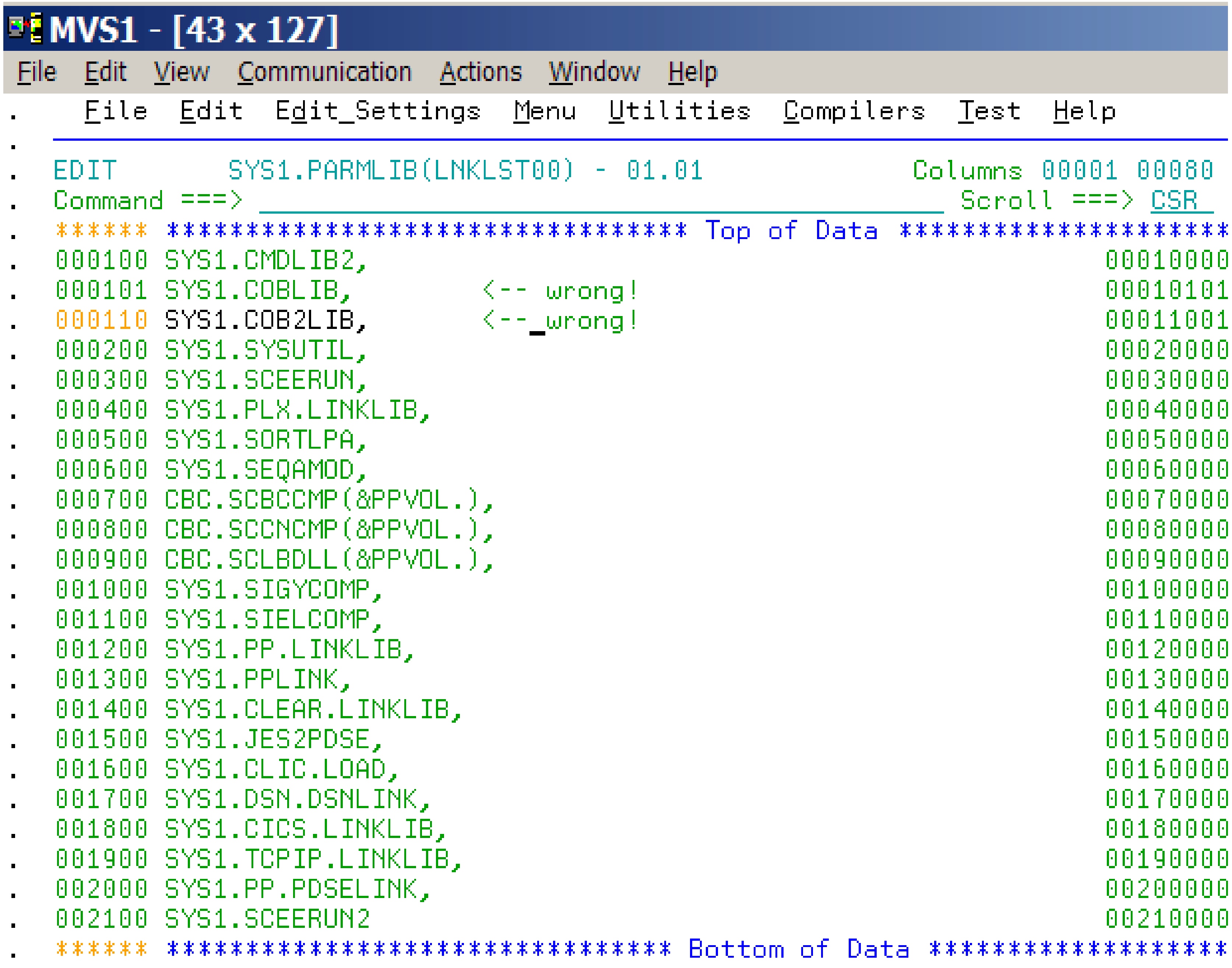


```
EDGE PORTFOLIO ANALYZER: COBNLEP V1R06.03 - 2006/04/04 13:29  PAGE    1
**** MODULES WITH COBOL TO MIGRATE TO ENTERPRISE COBOL             ****

                      PRE-                         LE CONF
 LOAD       COBOL    ANSV4  ANSV4  OS/VS   COBOL   COBOL   LIBR
 MODULE     CSECTS   COBOL  COBOL  COBOL    II    W/CMPR2   ID
 --------  --------  -----  -----  ------  -----  -------  ----
 BIPRT     BIPRT                             X             CASC
           WSIOB                           R2.0
           PRNTX       X
 CACN1     CACN1                           CMPR2           CASC
           WSIOB                           R2.0
 CACN2     CACN2                             X             CASC
           WSIOB                           R2.0
 GFSDJ     GFSDJ                             X             CASC
           CARESEGA                  X
           WSIOB                           R2.0
           PRNTY              X
           MBFR01                                  CMPR2
           CARESCML                  X
 KBXDD     KBXDD                   LVL(1)                  CASC
 KBXOA     CALLBACK                LVL(1)                  CASC

========================================================================

EDGE PORTFOLIO ANALYZER: COBNLEP V1R06.03 - 2006/04/04 13:29  PAGE   18
**** MODULES WITH COBOL TO MIGRATE TO ENTERPRISE COBOL             ****

TOTAL MODULES ANALYZED                                               575
TOTAL CSECTS  ANALYZED                                              3957

TOTAL OF ALL MODULES WITH COBOL CSECTS                               574
TOTAL OF ALL COBOL CSECTS                                            823
```

Preparation for Migration

- **Use only one run-time library for each application**
  - **Ex: COBOL NORES modules contain their own run-time routines**
  - **They must be REPLACED with link edit**
- **Do not have more than one COBOL library in LNKLST**
  - **If you have VS COBOL II ahead of Language Environment in LNKLST then you cannot do what IBM manuals say you can do**
    - ➤ Enterprise COBOL programs will not run
    - ➤ It is not supported
  - **IBM has received service calls caused by bad LNKLST setup**
- **For many shops, doing an LE migration will also be correcting an invalid LNKLST setup**

Use only one runtime library in LNKLST


MVS1 - [43 x 127]
File Edit View Communication Actions Window Help
File Edit Edit_Settings Menu Utilities Compilers Test Help

```
EDIT       SYS1.PARMLIB(LNKLST00) - 01.01               Columns 00001 00080
Command ===>                                               Scroll ===> CSR
****** ********************************* Top of Data **********************
000100 SYS1.CMDLIB2,                                              00010000
000101 SYS1.COBLIB,        <-- wrong!                             00010101
000110 SYS1.COB2LIB,       <--_wrong!                             00011001
000200 SYS1.SYSUTIL,                                              00020000
000300 SYS1.SCEERUN,                                              00030000
000400 SYS1.PLX.LINKLIB,                                          00040000
000500 SYS1.SORTLPA,                                              00050000
000600 SYS1.SEQAMOD,                                              00060000
000700 CBC.SCBCCMP(&PPVOL.),                                      00070000
000800 CBC.SCCNCMP(&PPVOL.),                                      00080000
000900 CBC.SCLBDLL(&PPVOL.),                                      00090000
001000 SYS1.SIGYCOMP,                                             00100000
001100 SYS1.SIELCOMP,                                             00110000
001200 SYS1.PP.LINKLIB,                                           00120000
001300 SYS1.PPLINK,                                               00130000
001400 SYS1.CLEAR.LINKLIB,                                        00140000
001500 SYS1.JES2PDSE,                                             00150000
001600 SYS1.CLIC.LOAD,                                            00160000
001700 SYS1.DSN.DSNLINK,                                          00170000
001800 SYS1.CICS.LINKLIB,                                         00180000
001900 SYS1.TCPIP.LINKLIB,                                        00190000
002000 SYS1.PP.PDSELINK,                                          00200000
002100 SYS1.SCEERUN2                                              00210000
****** ******************************** Bottom of Data ********************
```

Moving LE into Production

- Install target depends on migration strategy used
  - LNKLST/LPA only when everything runs under LE
- There are 2 main strategies being used, both OK:
  - Change applications one by one to use new compiler and run-time library (STEPLIB)
  - Run current modules under LE, then use new compiler later at maintenance or enhancement time (LNKLST)
- This presentation is focused on run-time first, then compiler migration
- STEPLIB for old applications that will not run under LE
  - Complete the migration later
- STEPLIB maximum control with performance cost
  - Customer feedback says STEPLIB noticeably slower
- One IMS region at a time
- One CICS region at a time

Moving LE into Production

- **Possible scenarios for controlled migration**
  - **Batch 1:**
    - ➢ Newly migrated applications STEPLIB to LE
    - ➢ Deferred applications STEPLIB to old runtime
    - ➢ At a given point change from old runtime to new in LPA/LNKLST
    - ➢ Start deleting STEPLIB for migrated applications
  - **Batch 2:**
    - ➢ Change all applications to STEPLIB to current runtime
    - ➢ Install LE in LPA/LNKLST
    - ➢ Delete STEPLIB to migrate each application

- **Scenario for controlled migration**
  - **Online:**
    - ➢ Create new region that uses LE
    - ➢ Move transaction by transaction from old region to new
  - **Online (TSO):**
    - ➢ Use TSOLIB command to set up dynamic steplib concatenation (must be TSO/E 2.5 or later)
    - ➢ Under ISPF add LE to the ISPLLIB concatenation
    - ➢ If you can control the logon proc, put a STEPLIB there

Moving LE into Production

- Some customers have used a bold migration method:
  - Test cross-section of applications
  - No or few problems were found
  - Dynamically change LNKLST on production systems on Sunday to have SCEERUN as the run-time library
  - Have all application support personnel on call for next week
  - STEPLIB applications that have problems
  - Fix the problems as time permits, remove STEPLIB
  - Method is not recommended as best method, but has worked for some

LE is Different!

- **Some run-time options are the same, many new ones**
- **ABEND codes are different**
  - **Most abends end up with code 4038**
  - **If TERMTHDACT(UADUMP) then 4039 also**
  - **Can get U1035-type codes with CEEWUCHA or CEEBX05A**
- **Run-time messages are in different places under CICS**
  - **CESE Transient Data queue**
    - VS COBOL II used Temporary Storage Queue
- **Other languages can be sub programs with LE**
  - **Example: COBOL calling C, C was MAIN before LE**
- **ABEND-AID installed differently than pre-LE**
- **Debug abends using error messages, NOT abend codes.**
  - **Abend codes will not help in many cases!**

LE is Different!

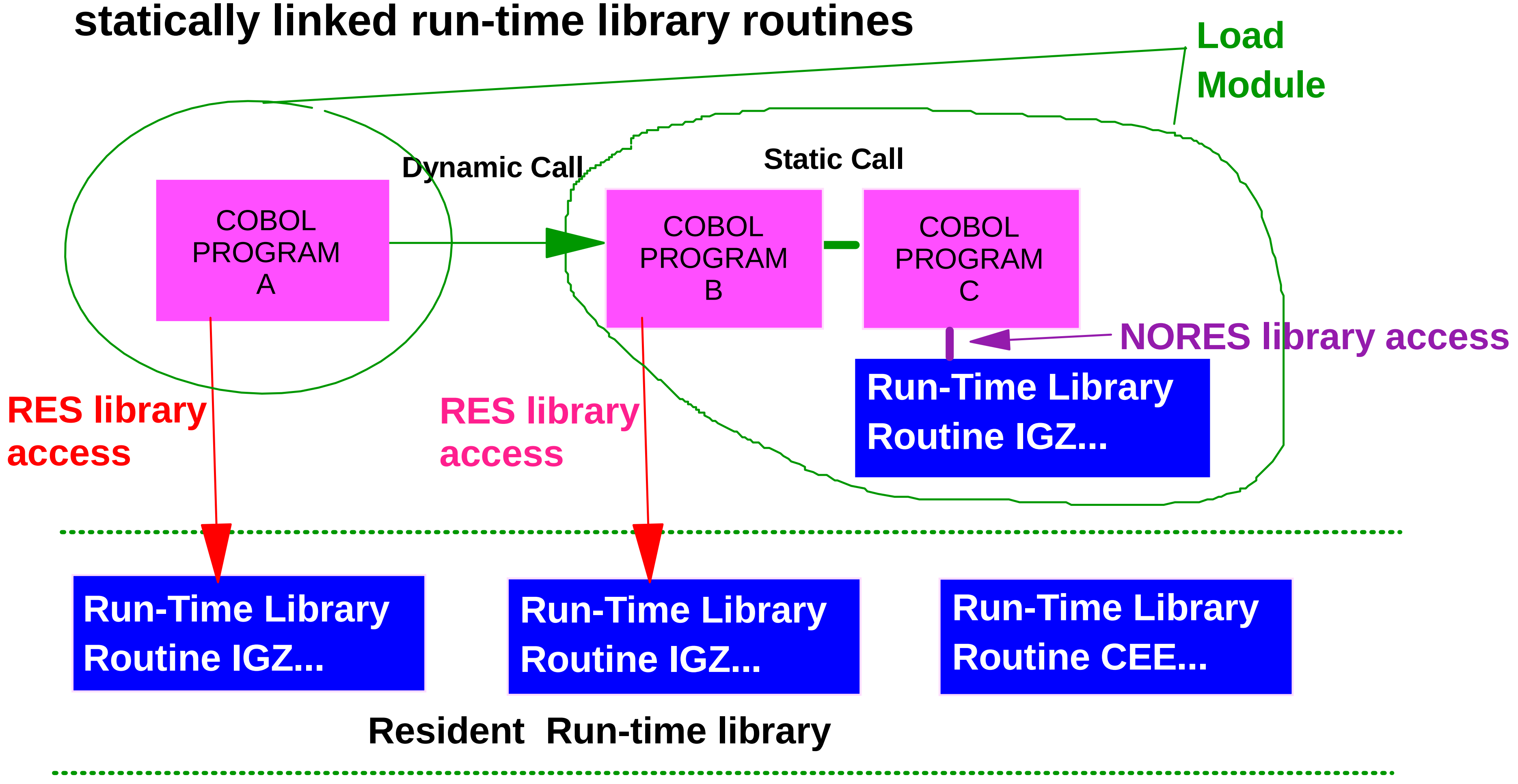
- LE supports downward compatiblity!
  - You can now develop applications on z/OS R7 and run on earlier releases of z/OS, such as R4
  - Supports the typical installation with higher level of OS on development system than production systems
- Old way: Save old LE and application build parts when installing a new release of z/OS
- Old Way: Build with z/OS R4 parts when z/OS R4 is the production level
- Old Way: See INFO APAR II11316 and web page URL: http://www.s390.ibm.com/le/assist/support/ii11316.html
- Current way (since OS/390 R10): Downward compatibility!

64-bit is not Different!

- PL/I and COBOL applications will run fine in 64-bit z/OS
  - 31-bit or 64-bit regions, it doesn't matter
  - Even when z/OS is running in z/Architecture (64-bit) mode
- Although PL/I and COBOL don't support 64-bit addresses explicitly, you may get some of the benefits of 64-bit z/OS just by moving to it. With 64-bit addressable real memory backing virtual memory, there will be less paging and swapping and the possibility of better system performance.
  - Actual results obtained in specific operating systems environments will vary depending on individual configurations and workload conditions.
- In addition, DB2 can exploit 64-bit addressing for SQL statements in PL/I and COBOL programs without any changes to the PL/I or COBOL programs
- Exploit 64-bit z/OS with AMODE 24 programs!
  - Of course, much better to be AMODE 31

RES/NORES vs DYNAM/NODYNAM

The difference between statically linked COBOL programs and statically linked run-time library routines


Load Module
Dynamic Call
Static Call
COBOL PROGRAM A
COBOL PROGRAM B
COBOL PROGRAM C
NORES library access
Run-Time Library Routine IGZ...
RES library access
RES library access
Run-Time Library Routine IGZ...
Run-Time Library Routine IGZ...
Run-Time Library Routine CEE...
Resident Run-time library

RES and NORES

- **All LE-conforming COBOL compilers are RES-only**
- **There is no MIXRES run-time option in LE**
  - **Mixed RES and NORES was common due to:**
    - ➤ Default compiler option was NORES
    - ➤ CALL identifier-1 forced RES
- **Mixed RES and NORES programs <u>are</u> supported in LE**
  - **Must relink with LE (or recompile)**
- **If mixed RES and NORES, then choose your approach in advance (one of the following):**
  - **Set up a separate environment for migrated applications and link edit with LE as you move each one**
  - **Use EDGE Portfolio Analyzer to build linkage editor control cards to relink many applications on the same day**
  - **Recompile all programs with RES before starting the migration to LE**

COBOL Source Conversion

COBOL Source Conversion

- After run-time migration complete, and run time has PTFs required for new compilers, think about compilers
- IBM recommends that you recompile with new compiler when:
  - Maintenance is done on an application
  - New function is added to an application
  - Any other program in a module is changed
- This approach provides a FREE compiler upgrade path
  - The code would be checked out anyway
  - The code would be tested before being promoted anyway
- IBM recommends that you use only one compiler
  - Recommend the newest one (V4R1 today)
  - Simplifies application maintenance
  - Simplifies application development tools and JCL
  - Saves money in compiler license fees
- You might have to keep the old compiler for emergency service

COBOL compiler migration

- **When planning to use a new compiler, follow these steps**
  - **Order PTFs for Language Environment required by new compiler**
  - **Install them in all systems where you run COBOL programs**
    - ➤ (This may take some time! Test, timing, etc)
  - **Order the new compiler**
  - **Install compiler and make available to programmers**
- **Alternative**
  - **Order PTFs for Language Environment required by new compiler**
  - **Install them on system where you want to run programs compiled by the new COBOL compiler**
  - **Order the new compiler**
  - **Install compiler and give restricted availability**
  - **Have a plan to get required PTFs on production systems**

COBOL Source Conversion

- **OldBOL to NewBOL**
  - **OS/VS COBOL had LOTS of undocumented extensions**
  - **85 Standard incompatible with 74 Standard**

Use Tools!

- **IBM COBOL Conversion Aid included in Debug Tool**
  - **If you buy Debug Tool Utilities and Advanced Functions(DTUAF)**
    - ➤DTUAF V6 5655-P15
    - ➤DTUAF V7 5655-R45
    - ➤DTUAF V8 5655-S16
- **IBM COBOL and CICS Command Level Conversion Aid**
  - **Over 80 user requirements added in Version 1 Release 2**
  - **Current product is CCCA Version 2**
    - ➤More improvements over Version 1 Release 2
  - **Now a full Program Product:**

    5648-B05 (z/OS, VM)

    5686-A07 (VSE/ESA)

What is the CCCA Conversion Aid?

- **Converts source programs written for:**
  - **OS/VS COBOL**
  - **DOS/VS COBOL**
  - **VS COBOL II (or later) w/CMPR2**
- **Into current IBM COBOL compiler syntax**
  - **Programs that can be compiled by:**
    - ➤Enterprise COBOL for z/OS
    - ➤COBOL for OS/390 & VM
    - ➤COBOL for MVS & VM
    - ➤COBOL for VSE/ESA
    - ➤VisualAge COBOL (Windows and OS/2)
    - ➤COBOL Set for AIX
    - ➤ILE COBOL for AS/400
    - ➤VS COBOL II w/NOCMPR2

What about compiler compatibility?

- Once your programs are converted to 85 Std you can change compilers at any time
  - VS COBOL II NOCMPR2
  - COBOL for MVS & VM NOCMPR2
  - COBOL for OS/390 & VM NOCMPR2
  - Enterprise COBOL V3 and V4
- You can compile any existing programs at any time with a newer compiler with guaranteed same results!
  - Migrate gradually at service/update time
  - No need to convert VS COBOL II NOCMPR2 programs, just use the new compiler

Summarize new reserved words

- **New reserved words by compiler, compared to VS COBOL II:**
  - **COBOL/370 V1R1**
    - ➤ FUNCTION, PROCEDURE-POINTER
  - **COBOL for MVS & VM V1R2**
    - ➤ CLASS-ID LOCAL-STORAGE OBJECT RETURNING
    - ➤ END-INVOKE METACLASS OVERRIDE SELF
    - ➤ INHERITS METHOD RECURSIVE SUPER
    - ➤ INVOKE METHOD-ID REPOSITORY
  - **COBOL for OS/390 & VM V2R1 (no new words)**
  - **COBOL for OS/390 & VM V2R2**
    - ➤ COMP-5 COMPUTATIONAL-5 FACTORY
    - ➤ END-EXEC EXEC SQL TYPE
    - ➤ (FACTORY added to list of words NOT reserved with NOOO)
  - **COBOL for OS/390 & VM V2R2 w/PQ45462**
    - ➤ EXECUTE

Summarize new reserved words

- New reserved words in Version 3:
  - JNIENVPTR, NATIONAL, XML, END-XML, XML-EVENT, XML-CODE, XML-TEXT, XML-NTEXT, FUNCTION-POINTER
  - WORD(NOOO) compiler option no longer available
- New reserved words for V3R4:
  - GROUP-USAGE, NATIONAL-EDITED

Compiler Migration New News

- **Enterprise COBOL Version 4 Release 1!**
- **Run-time library pre-reqs:**
  - **z/OS R7, R8 or R9 Language Environment**
    - ➤Plus PTFs for APAR PK55645
- **New reserved words for V4R1:**
  - **AS, ATTRIBUTES, ELEMENTS, ENCODING, NAMESPACE, NAMESPACE-PREFIX, XML-DECLARATION, XML-NAMESPACE, XML-NAMESPACE-PREFIX, XML-NNAMESPACE, XML-NNAMESPACE-PREFIX**
- **CCCA is updated for reserved word conversions for Enterprise COBOL Version 4 Release 1 by the PTF for APAR PK57361.**

Enterprise COBOL

- Enterprise COBOL is 1985 Standard only
  - CMPR2 option has been removed
  - You can use Debug Tool (CCCA) to easily convert
- Java-based OO only
  - SOM-based OO COBOL has been removed
    - ➤ IDLGEN and TYPECHK options removed
  - SOM was removed from z/OS V1R2
- New default options:
  - DBCS   FLAG(I,I)   RENT XREF(FULL)

COBOL Compiler Options

- **OPT(FULL)**
  - **Don't use for programs with flags in WORKING-STORAGE unless you add references to them**
  - **The OPTIMIZER will delete a dead reference anyway, but OPT(FULL) will not delete the data item if it is referenced**

```
77 BEG-MARKER  PIC X(20) VALUE "BEGINNING OF W-S".
 ...
 ...
IF BEG-MARKER NOT EQUAL BEG-MARKER THEN
   DISPLAY BEG-MARKER.
```

COBOL Compiler Options

- **If currently OS/VS COBOL using TRUNC as default or VS COBOL II using TRUNC(STD) as default:**
  - Use TRUNC(STD) as default under Enterprise COBOL
  - They are identical!
- **If currently OS/VS COBOL using NOTRUNC as default:**
  - Use TRUNC(OPT) as default under Enterprise COBOL
  - They are very similar
- **If currently VS COBOL II Release 2 using NOTRUNC as default:**
  - Use TRUNC(OPT) as default under Enterprise COBOL
  - They are identical!
- **TRUNC(BIN) should NOT be the default option**
  - It is TOO SLOW
  - Not as slow as it used to be!
- **TRUNC(BIN) should be used only for select programs that require guaranteed non-truncation of binary data**
  - This is particularly true if there is a possibility that data being moved into BINARY data items can have a value larger than that defined by the PICTURE clause for the BINARY item

Adding New COBOL to Old COBOL Applications

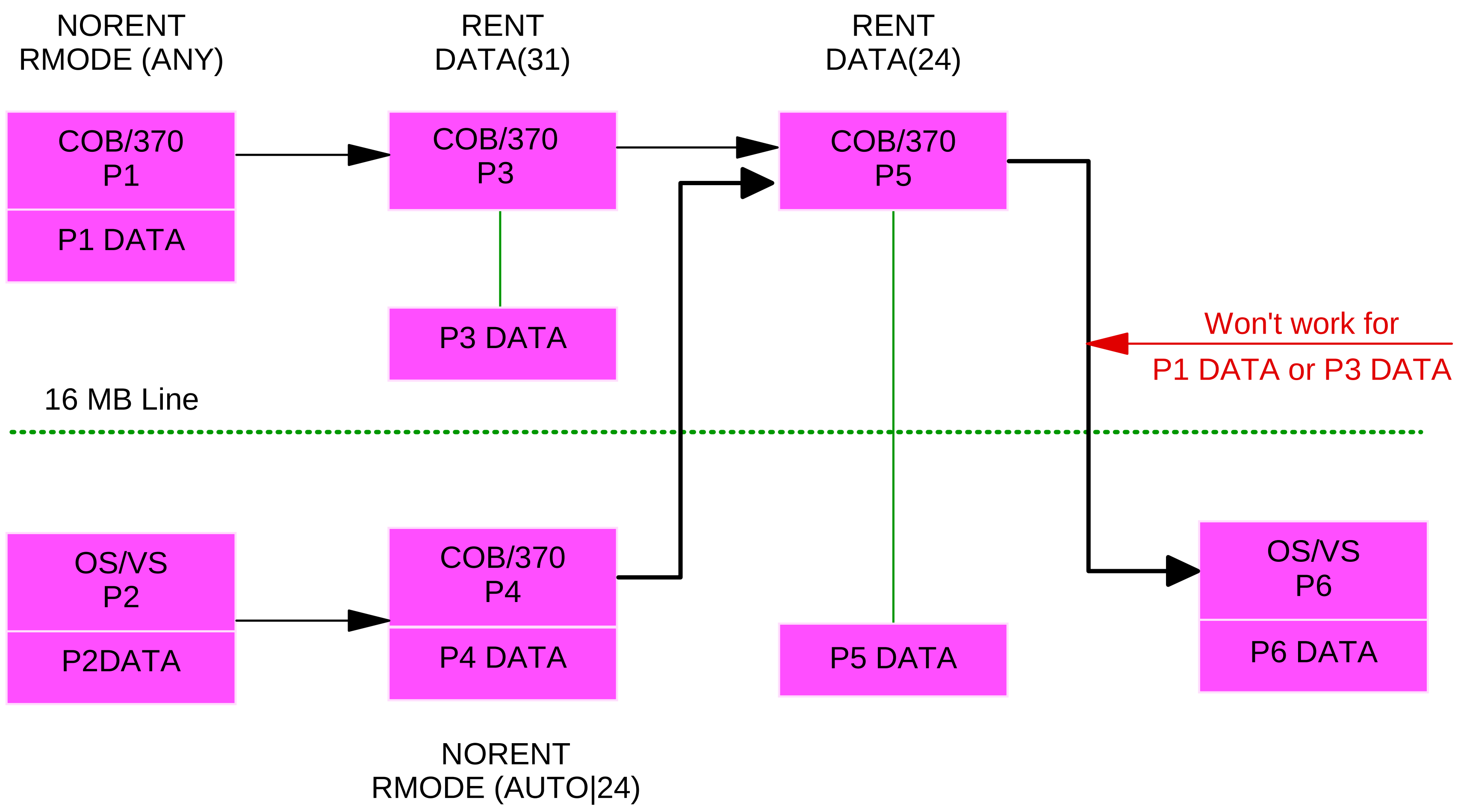
- **Example: Changing one program to Enterprise COBOL in an OS/VS COBOL application**
  - **A little about AMODE and RMODE**
    - ➢ OS/VS programs are all RMODE=24 AMODE=24
    - ➢ Enterprise COBOL programs can be RMODE=24 or RMODE=ANY
    - ➢ Enterprise COBOL programs are all AMODE=ANY
    - ➢ Linkage Editor/Binder can override
  - **Are you ready for the 16 MB line?**

Adding New COBOL to Old COBOL Apps

- Keep all data below 16MB until all source migrated
  - If RENT, then DATA(24)
  - If NORENT, RMODE(24) or RMODE(AUTO)
- I recommend migrating bottom subroutines first
- Old COBOL calling new COBOL is safer than new calling old
- Dynamic call is safer than static, mode-switching
  - Can't enter an OS/VS COBOL program with AMODE=31!
  - Get IKF995I ILBONTR AMODE ERROR

Adding New COBOL to Old Apps

Example: Mixing new and old (all calls DYNAMIC)



Adding New COBOL to Old COBOL Apps

- **Static calls from Enterprise COBOL programs to VS COBOL II programs**
  - **Must use up-to-date copy of IGZEBST in the load module:**
    - ➤IGZEBST from any OS/390 or z/OS LE
    - ➤ IGZEBST from pre-OS/390 LE R5 (or later) SCEELKED library
    - ➤ IGZEBST from pre-OS/390 LE R3 or R4 SCEELKED w/PN74011 applied
    - ➤ IGZEBST from VS COBOL II Release 4 library with PN74000 applied
  - **Documented in Chapter 18 of GC27-1409 Migration guide: 'Adding Enterprise COBOL Programs to existing applications'**
- **Language Environment user exits**
  - –
  - –

Hints and Tips

- Have at least one "COBOL DBA" who knows about compilers, run-times, and preferred default settings
  - The go-to person for COBOL/LE product questions
- Keep your language products up-to-date on service as much as possible
  - Down-level language migrations are harder
  - Example: PN74000 for VS COBOL II Release 4 would prevent problems with IGZEBST when adding COBOL for MVS & VM programs to VS COBOL II applications
- Keep control of your language products usage
  - Do not allow programmers to choose the release and maintenance level or you will end up with a nightmare for application management

Preparation for Migration

- Do not have more than one COBOL library in concatenation
- If more than one COBOL library in concatenation
  - If LE is first, then other COBOL is dead code
  - If VS COBOL II is first, then OS/VS COBOL is dead
- Common error we are hearing:
  - VS COBOL II in LPA, then LE in LNKLST
  - This will cause problems! It is NOT supported!
  - VS COBOL II first means the LE migration still has to be done
- If LE not first, what could go wrong?
  - Mixing COBOL for OS/390 & VM programs and VS COBOL II programs will definitely fail, but in unpredictable ways
  - We all like mystery problems at 3:00 AM, right?
- One COBOL library in concatenation only
  - Either COBLIB, VSCLLIB, COB2LIB, or SCEERUN
  - COBLIB and VSCLLIB not supported
  - COB2LIB support ended 3/2001

Inventory of applications

The most difficult migrations

- OS/VS COBOL NORES programs that use Assembler with SVC LINK for "CALLing" subprograms
  - SVC LINK creates new enclave, OS/VS under LE must be single enclave in non-CICS
- Any applications that do their own program management with assembler
- Shops that use exits on the DCB EXLST for label processing
  - VS COBOL II and COBOL for MVS & VM use them all
- Any shop using modified ILBO* modules
- Applications that set SPIEs or STAEs and trap abends
  - Must migrate to LE condition handling
  - We have some relief for VS COBOL II programs that use SPIE under LE, as long as SPIE is RESET after application code
- OS/VS COBOL ILC with PL/I
  - Must upgrade COBOL before migrating to Language Environment

Technical Advantages of LE

Performance

- **Inter-language CALLs are faster under LE between COBOL and other languages, with about 80% less overhead for each call**
- **COBOL-assembler-COBOL calls have 80% less overhead than VS COBOL II**
- **All COBOL calls are faster under LE than under VS COBOL II**
- **In some cases, DYNAMIC CALL is faster than under OS/VS COBOL!**
- **Reusable run-time environment for PL/I programs now available under IMS!**

Technical Advantages of LE

Performance

- COBOL compiler changes for object program performance (Improvements over VS COBOL II):
  - Vastly improved generated code for TRUNC(BIN) w/BINARY math
  - OPTIMIZE(FULL) compiler option
  - Static initialization of WORKING-STORAGE variables at compile time for NORENT programs
  - Optimized parameter list generated code (for USING phrase)
  - Improved performance of variable-length MOVEs
  - Optimized code generated for main entry point
  - Dynamic CALL literal performance improved

Technical Advantages of LE

New function

- **Support for z/OS Unix System Services, including support for some of the C language functions defined in the POSIX standard**
- **ANSI 1985 Intrinsic Functions available in COBOL for MVS & VM, COBOL for OS/390 & VM, and Enterprise COBOL**
- **High-precision math routines available to any language**
  - **LE callable services are only available to LE-conforming languages or LE-conforming assembler**

Technical Advantages of LE

New function

- LE condition handling (abend intercept, "0C7 catcher", etc.)

  Allows application-level condition handling so that each routine can be written without dealing with error conditions. This allows routines to be written for success and performance rather than written to address errors in individual statements. The error-handling routines can be written in the High Level Language of choice, thus helping you move away from assembler.

  (Before LE the error-handling code had to be in the same language as the application, or in assembler language. Also, in COBOL you could only have statement-specific condition handling, which means each new program had to have its own condition handling and the associated performance overhead.)

Technical Advantages of LE

New function

- **Enterprise COBOL, Enterprise PL/I**
- **Object-Oriented COBOL on z/OS requires LE**
- **Client-server using COBOL on z/OS requires LE**
- **C++ and Java for z/OS require LE**
- **DB2 Stored Procedures require LE**
- **CICS Autoinstall Exit written in COBOL requires LE**

Technical Advantages of LE

Platform of the future

- **Data items larger than 16MB, up to 128 MB, only with Enterprise COBOL V3R4 or V4R1**
- **31-digit decimal data items only in Enterprise COBOL and COBOL for OS/390 & VM, or Enterprise PL/I for z/OS**
- **Any future performance improvements or language enhancements such as WebSphere support will only be available under LE.**
- **LE is an element of z/OS. LE date/time services are available on Windows, OS/2 and AIX as well. LE is already built into the OS/400 operating system.**
- **z/OS UNIX support for C and COBOL requires LE.**

Technical Advantages of LE

System exploitation

- QSAM buffers can now be above the line for full exploitation of DFSMS and data striping.
- COBOL EXTERNAL data can now be above the 16 MB line for COBOL for MVS & VM
  - (Had to be below the line with VS COBOL II programs)
- CBLPSHPOP run-time option allows choice of performance versus error handling. (Under VS COBOL II the user had no choice and always took the performance hit of PUSH HANDLE and POP HANDLE for CALLs)
- COBOL compiler options:
  - RMODE option allows NORENT programs to run above the line
  - OPT(FULL) automatically optimizes out unused data items
- New LE support in R3:
  - Allows COBOL for MVS & VM programs compiled with NORENT and RMODE(ANY) to run above the 16 MB line. (Only RENT was supported above the line under VS COBOL II.)

Technical Advantages of LE

Usability

- **All language error messages now in latest CD-ROM collection kit (previous to LE, messages in 3 different manuals)**
- **CBLQDA run-time option prevents dynamic allocation of QSAM files**
  - **Only available as a User Mod under VS COBOL II:**
    - ➤ APAR II04562
    - ➤ Included ZAP and instructions
  - **CBLQDA works regardless of whether VS COBOL II User Mod was used**
- **Enterprise COBOL removes restrictions for APPLY WRITE-ONLY. (Under OS/VS COBOL, had to use FROM on WRITE. Also, the subfields of the records could not be referred to in procedure statements.)**
- **Storage tuning via run-time options (no link-edited assembler, which was the only way to tune storage usage under VS COBOL II)**