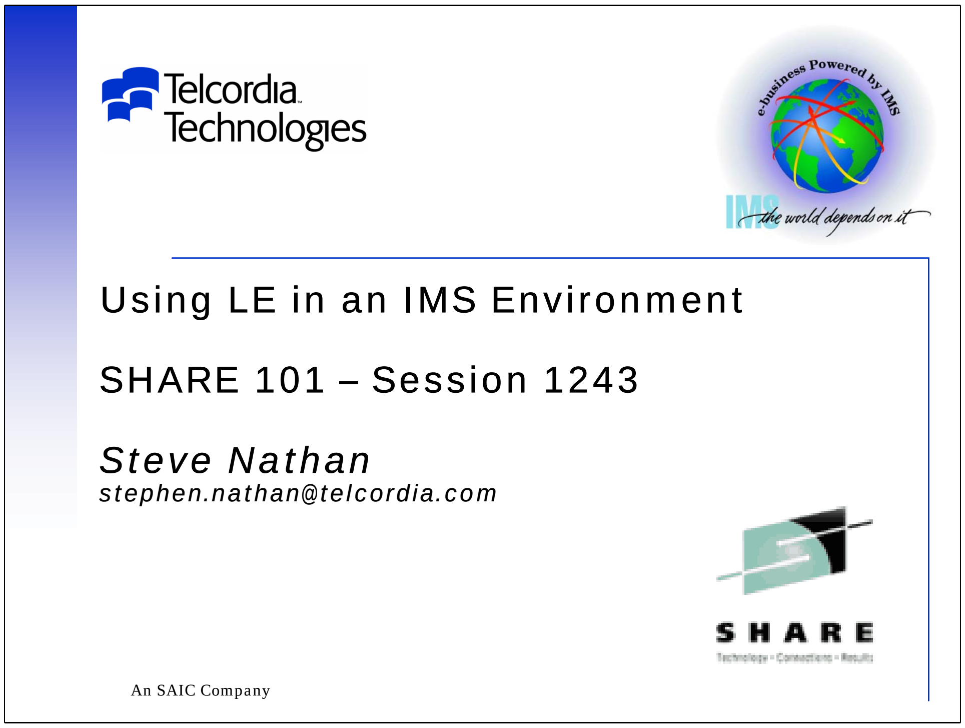


Telcordia Technologies

e-business Powered by IMS

IMS the world depends on it

Using LE in an IMS Environment

SHARE 101 – Session 1243

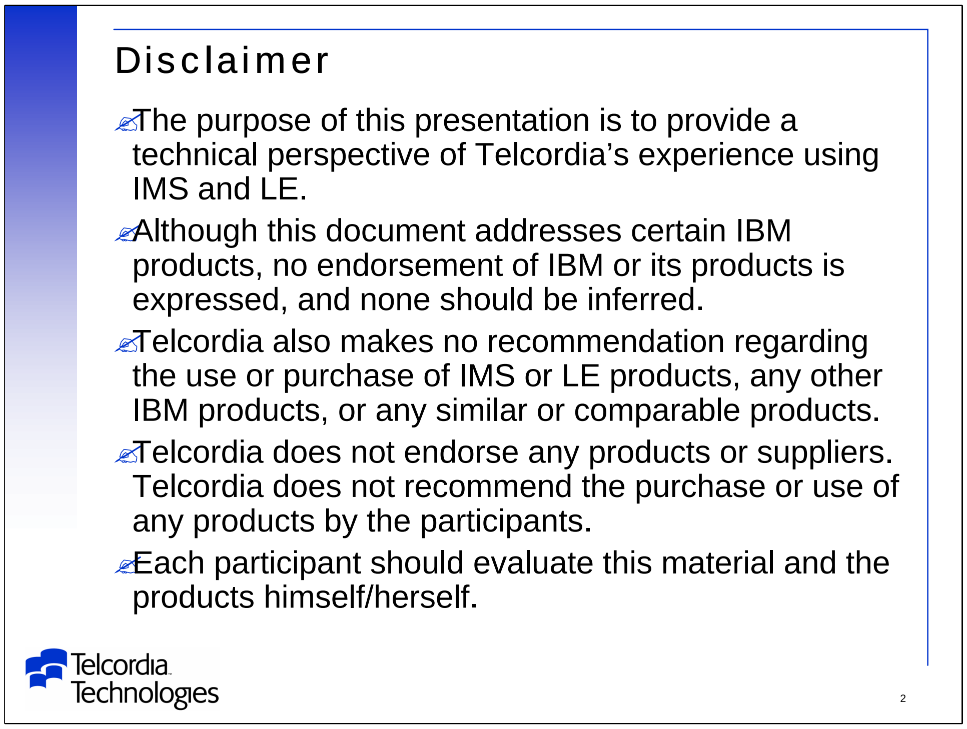
Steve Nathan
stephen.nathan@telcordia.com

SHARE
Technology • Connections • Results

An SAIC Company

Welcome to "Using LE in an IMS Environment"

Notes have been provided for your convenience.



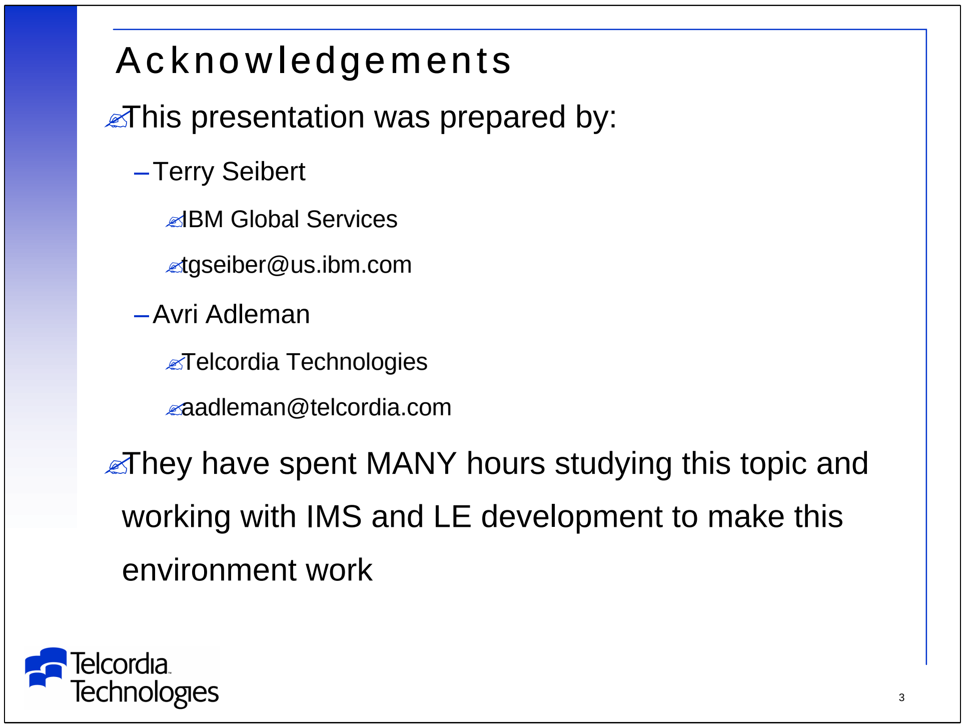
Disclaimer

- The purpose of this presentation is to provide a technical perspective of Telcordia's experience using IMS and LE.
- Although this document addresses certain IBM products, no endorsement of IBM or its products is expressed, and none should be inferred.
- Telcordia also makes no recommendation regarding the use or purchase of IMS or LE products, any other IBM products, or any similar or comparable products.
- Telcordia does not endorse any products or suppliers. Telcordia does not recommend the purchase or use of any products by the participants.
- Each participant should evaluate this material and the products himself/herself.

Telcordia Technologies

2

A legal requirement of Telcordia.



Acknowledgements

- This presentation was prepared by:
  - Terry Seibert
    - IBM Global Services
    - tgseiber@us.ibm.com
  - Avri Adleman
    - Telcordia Technologies
    - aadleman@telcordia.com
- They have spent MANY hours studying this topic and working with IMS and LE development to make this environment work

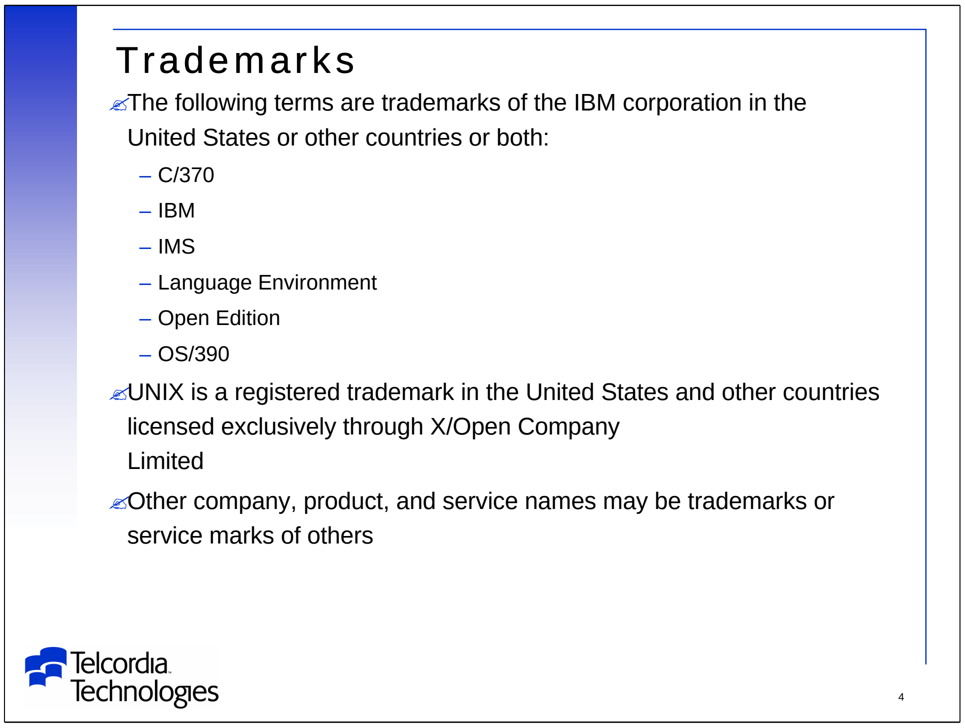
Telcordia Technologies

3

Terry Seibert and Avri Adleman get all of the credit for this presentation.

They should also get all of the credit for working together with IBM LE development to make LE and LE/IMS work.

There were MANY problems in the early days of LE.
- We had many meetings with the LE developers.



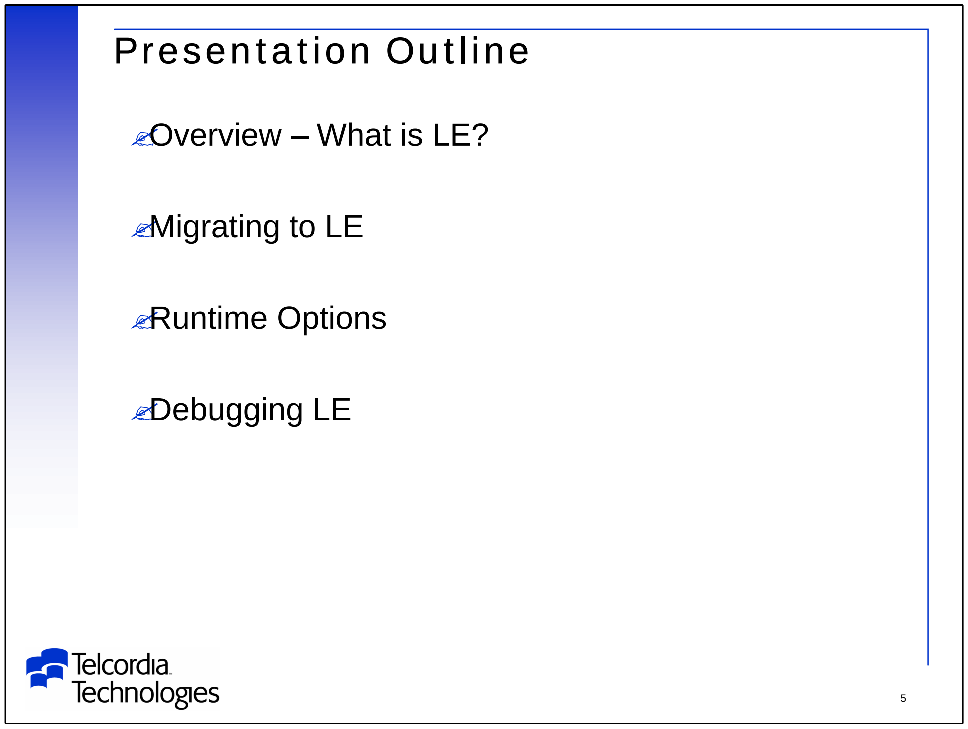
Trademarks

- The following terms are trademarks of the IBM corporation in the United States or other countries or both:
  - C/370
  - IBM
  - IMS
  - Language Environment
  - Open Edition
  - OS/390
- UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited
- Other company, product, and service names may be trademarks or service marks of others

Telcordia Technologies

4

A legal requirement of IBM.



Presentation Outline

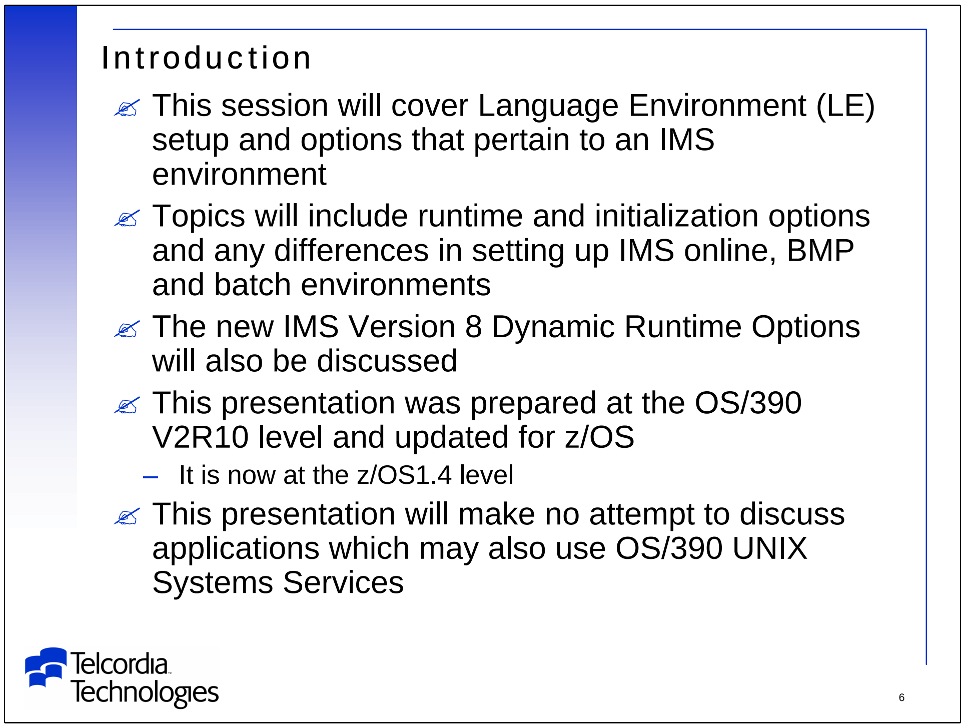
- Overview – What is LE?
- Migrating to LE
- Runtime Options
- Debugging LE

Telcordia Technologies

5

For those of us that do not know LE – an introduction is required.

Like any other system – LE has its own way of looking at things and naming things – even if they had perfectly good names before.



Introduction

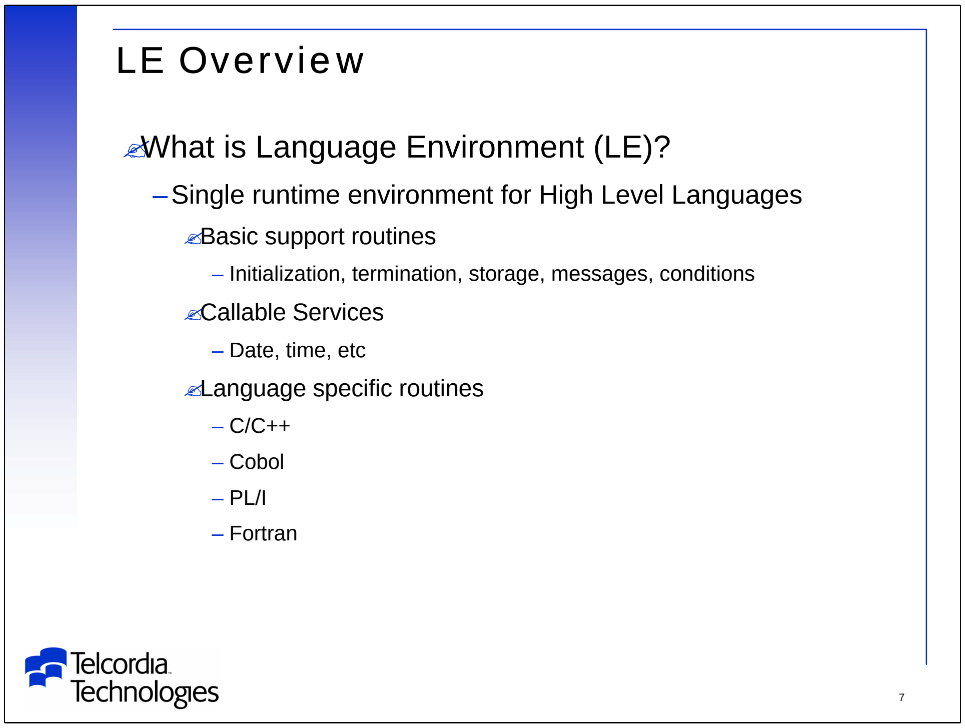
- ✍ This session will cover Language Environment (LE) setup and options that pertain to an IMS environment
- ✍ Topics will include runtime and initialization options and any differences in setting up IMS online, BMP and batch environments
- ✍ The new IMS Version 8 Dynamic Runtime Options will also be discussed
- ✍ This presentation was prepared at the OS/390 V2R10 level and updated for z/OS
  - It is now at the z/OS1.4 level
- ✍ This presentation will make no attempt to discuss applications which may also use OS/390 UNIX Systems Services

Telcordia Technologies

6

This presentation was originally prepared at the OS/390 V2R10 level.

It has been updated for z/OS 1.2 and z/OS 1.4.



LE Overview

- What is Language Environment (LE)?
  - Single runtime environment for High Level Languages
    - Basic support routines
      - Initialization, termination, storage, messages, conditions
    - Callable Services
      - Date, time, etc
    - Language specific routines
      - C/C++
      - Cobol
      - PL/I
      - Fortran

Telcordia Technologies

7

Language Environment (LE) consists of:

-Basic routines that support starting and stopping programs, allocating storage, communicating with programs written in different languages, and handling conditions.

-Callable services, such as math services and date/time services, that are commonly needed by programs

-Language-specific portions of the runtime library.

-Because many language-specific routines call LE services, behavior is consistent across languages.

LE runtime library includes the following routines:

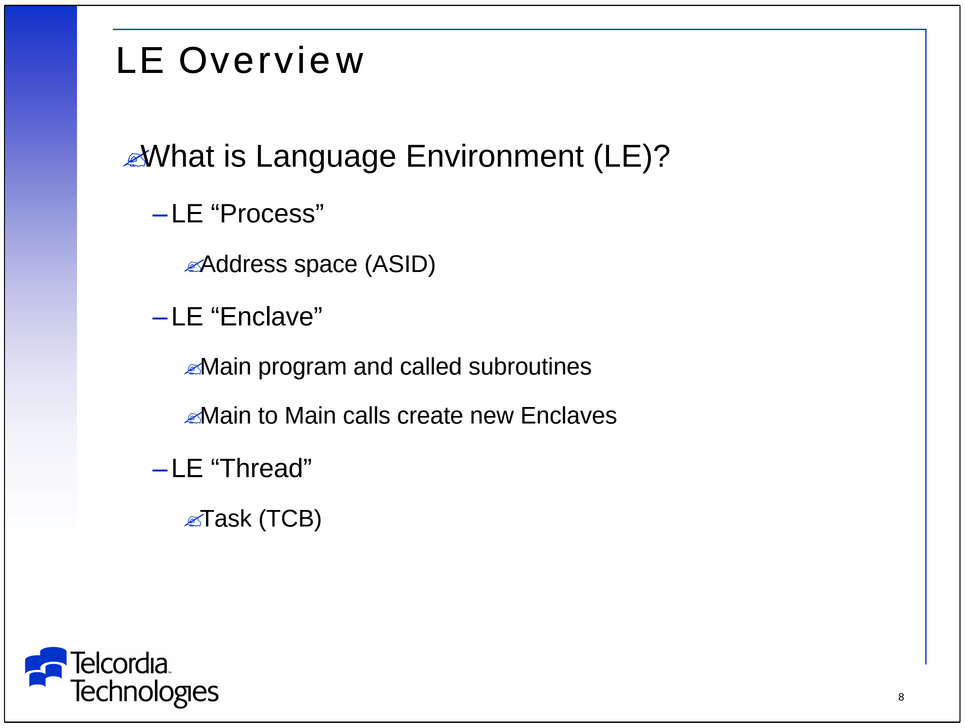
-CEExxxxx – Common routines

-EDCxxxxx – C/C++ specific routines

-IBMxxxxx – PL/I specific routines

-IGZxxxxx – Cobol specific routines

-AFHxxxxx – Fortran specific routines



LE Overview

- What is Language Environment (LE)?
  - LE “Process”
    - Address space (ASID)
  - LE “Enclave”
    - Main program and called subroutines
    - Main to Main calls create new Enclaves
  - LE “Thread”
    - Task (TCB)

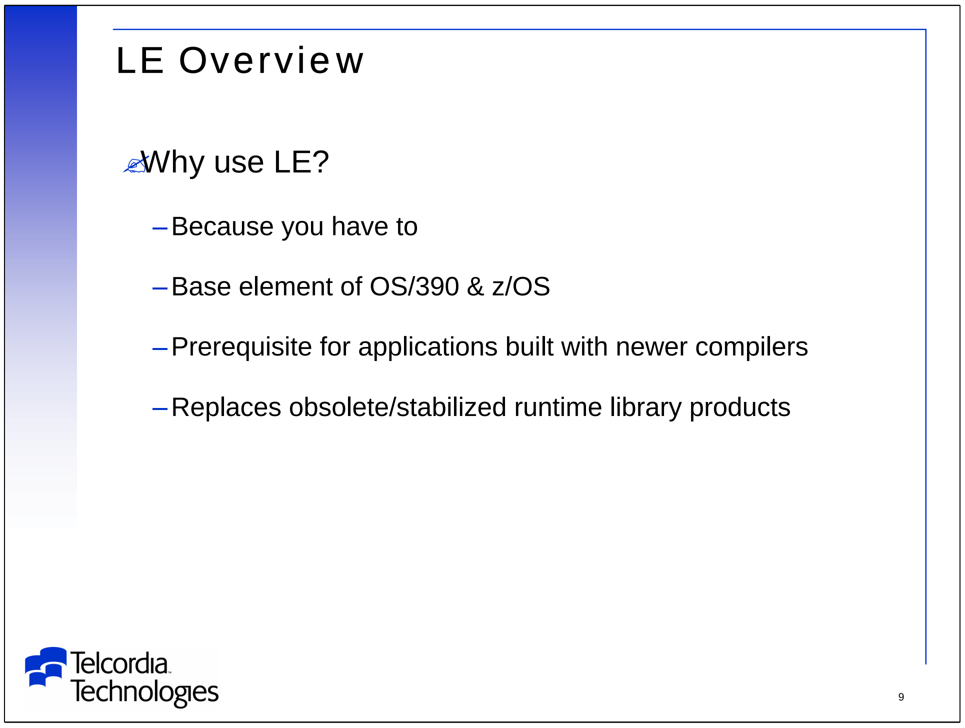
Telcordia Technologies

8

LE has new names for old concepts.

My definitions here are not exact – but they are simple and basically correct.

See the LE Concepts Guide for exact definitions.
-The manual is small and worth reading cover to cover.



LE Overview

- Why use LE?
  - Because you have to
  - Base element of OS/390 & z/OS
  - Prerequisite for applications built with newer compilers
  - Replaces obsolete/stabilized runtime library products

Telcordia Technologies

9

LE is prerequisite for applications built with newer compilers such as C/C++, Enterprise COBOL, and Enterprise PL/I.

LE replaces obsolete/stabilized (working?!?) runtime library products:

-OS/VS COBOL

-VS COBOL II

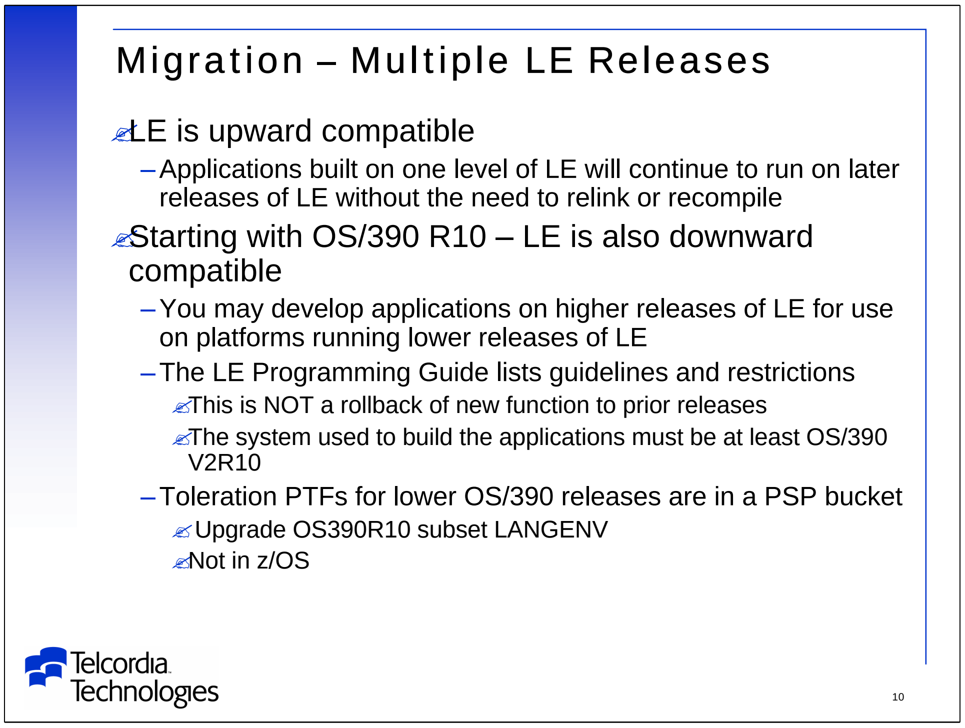
-OS/PL/I

-C/370

-OS FORTRAN

-VS FORTRAN

Read the LE Concepts Guide for an overview of LE and a description of the LE program model.



Migration – Multiple LE Releases

- LE is upward compatible
  - Applications built on one level of LE will continue to run on later releases of LE without the need to relink or recompile
- Starting with OS/390 R10 – LE is also downward compatible
  - You may develop applications on higher releases of LE for use on platforms running lower releases of LE
  - The LE Programming Guide lists guidelines and restrictions
    - This is NOT a rollback of new function to prior releases
    - The system used to build the applications must be at least OS/390 V2R10
  - Toleration PTFs for lower OS/390 releases are in a PSP bucket
    - Upgrade OS390R10 subset LANGENV
    - Not in z/OS

Telcordia Technologies

10

The downward compatibility support provided by OS/390 R10, and by the toleration PTFs, does not change LE's upward compatibility.

Applications developed exploiting the downward compatibility support must not use LE function that is unavailable on the lower release where the application will be deployed.

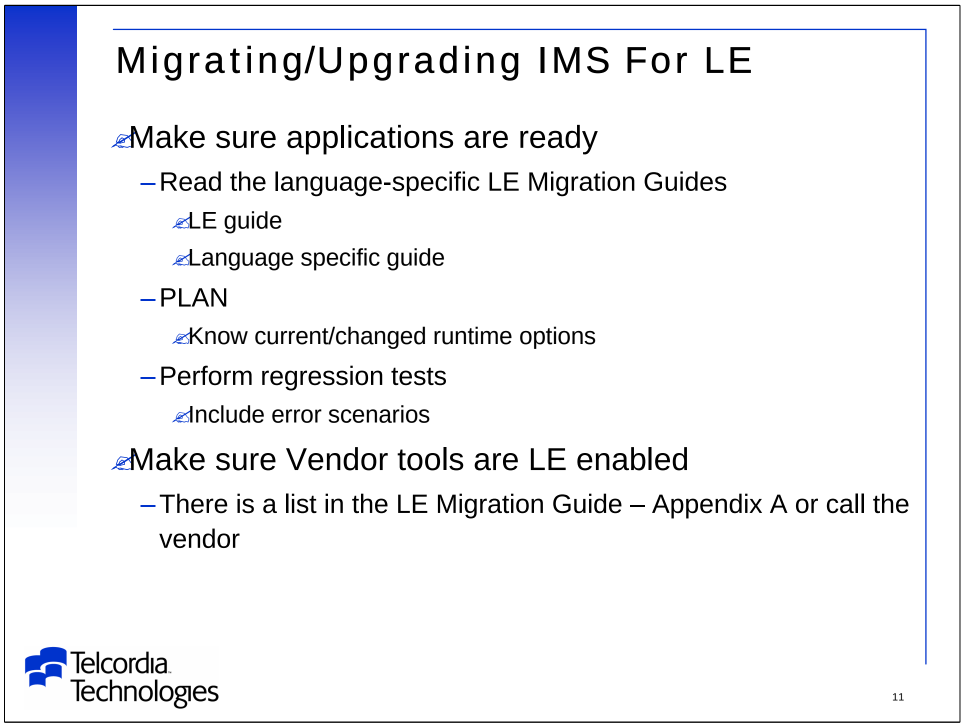
The downward compatibility support includes toleration PTFs for lower releases of LE to assist in the diagnosis of applications that violate the programming requirements for this support.

The diagnosis assistance that will be provided by the toleration PTFs includes:

-Options processing

-Detection of unsupported function

-C/C++ headers



Migrating/Upgrading IMS For LE

- Make sure applications are ready
  - Read the language-specific LE Migration Guides
    - LE guide
    - Language specific guide
  - PLAN
    - Know current/changed runtime options
  - Perform regression tests
    - Include error scenarios
- Make sure Vendor tools are LE enabled
  - There is a list in the LE Migration Guide – Appendix A or call the vendor

Telcordia Technologies

11

READ – With certain exceptions, LE provides object and load module compatibility for applications that are generated with many pre-LE IBM language products.

-Load modules that are created with these compilers and linked with their associated runtime libraries will run compatibly with LE without relinking.

-Also, object modules created with these compilers can be linked and run with LE without recompiling – but may uncover old errors.

- Storage and save areas are acquired and processed differently with LE.

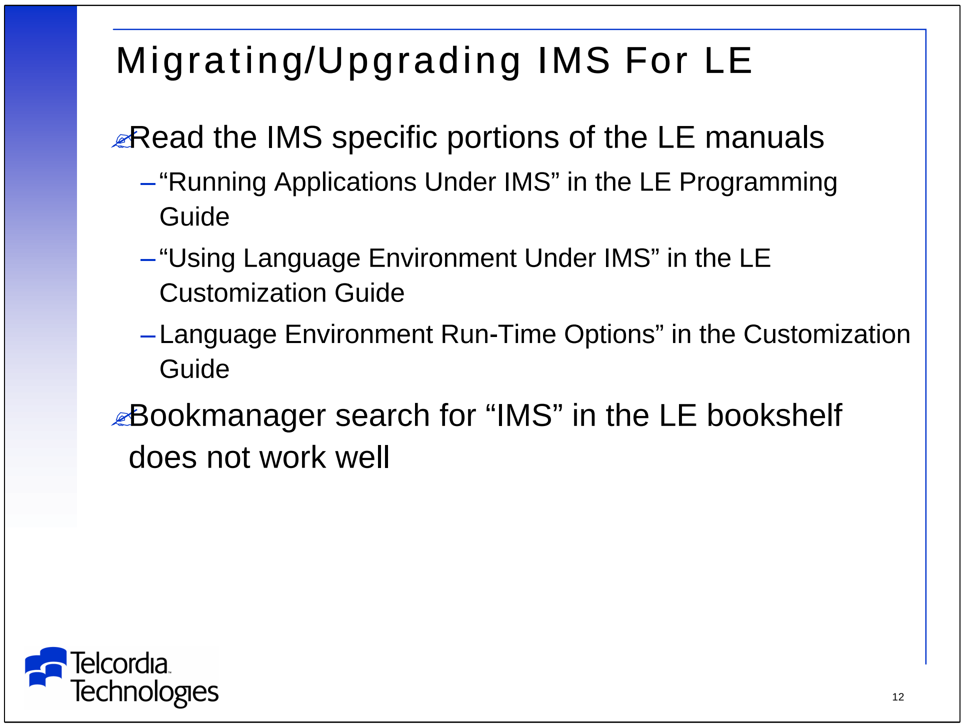
PLAN – Ensure that you and other application programmers who will be involved in the migration effort are familiar with the features of LE.

-You should be aware of the differences between your current runtime environment and options and the environment and options that LE provides.

PLAN – Take an inventory of the applications, vendor products, and tools you intend to run with LE.

-Know what version and release of which compiler generated the program and which runtime options were used and how they were specified.

PLAN – Determine the effort required to migrate each program and the order in which you will migrate them.



Migrating/Upgrading IMS For LE

- Read the IMS specific portions of the LE manuals
  - “Running Applications Under IMS” in the LE Programming Guide
  - “Using Language Environment Under IMS” in the LE Customization Guide
  - Language Environment Run-Time Options” in the Customization Guide
- Bookmanager search for “IMS” in the LE bookshelf does not work well

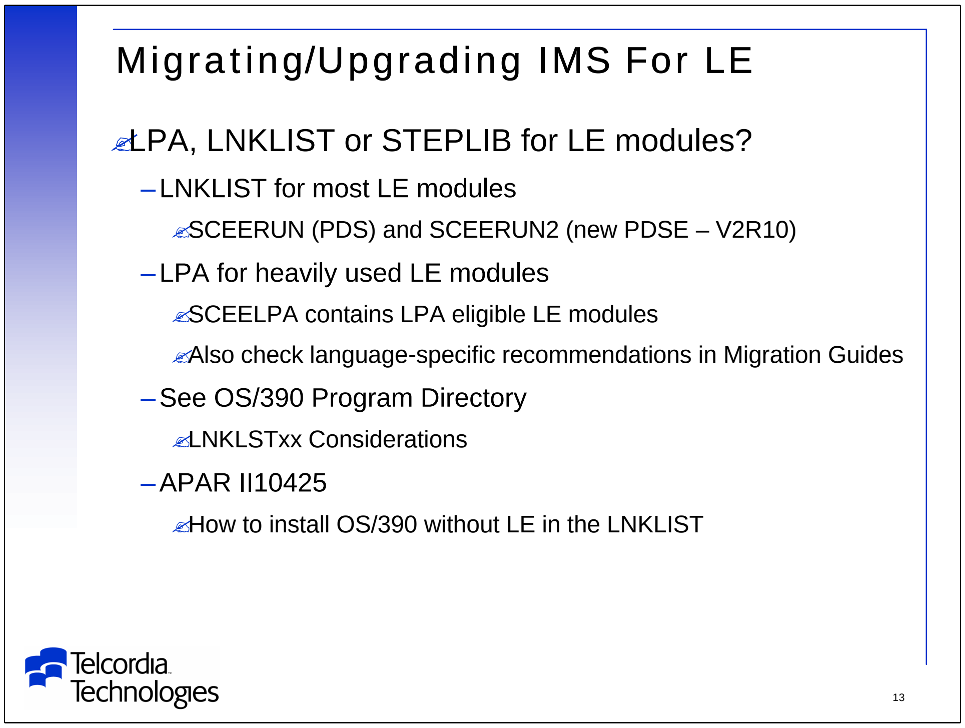
Telcordia Technologies

12

The LE manuals have several sections dealing specifically with IMS.

There are many other references to IMS in the LE manuals.

- Bookmanager search is not great at finding these references.



Migrating/Upgrading IMS For LE

- LPA, LNKLIST or STEPLIB for LE modules?
  - LNKLIST for most LE modules
    - SCEERUN (PDS) and SCEERUN2 (new PDSE – V2R10)
  - LPA for heavily used LE modules
    - SCEELPA contains LPA eligible LE modules
    - Also check language-specific recommendations in Migration Guides
  - See OS/390 Program Directory
    - LNKLSTxx Considerations
  - APAR II10425
    - How to install OS/390 without LE in the LNKLIST

Telcordia Technologies

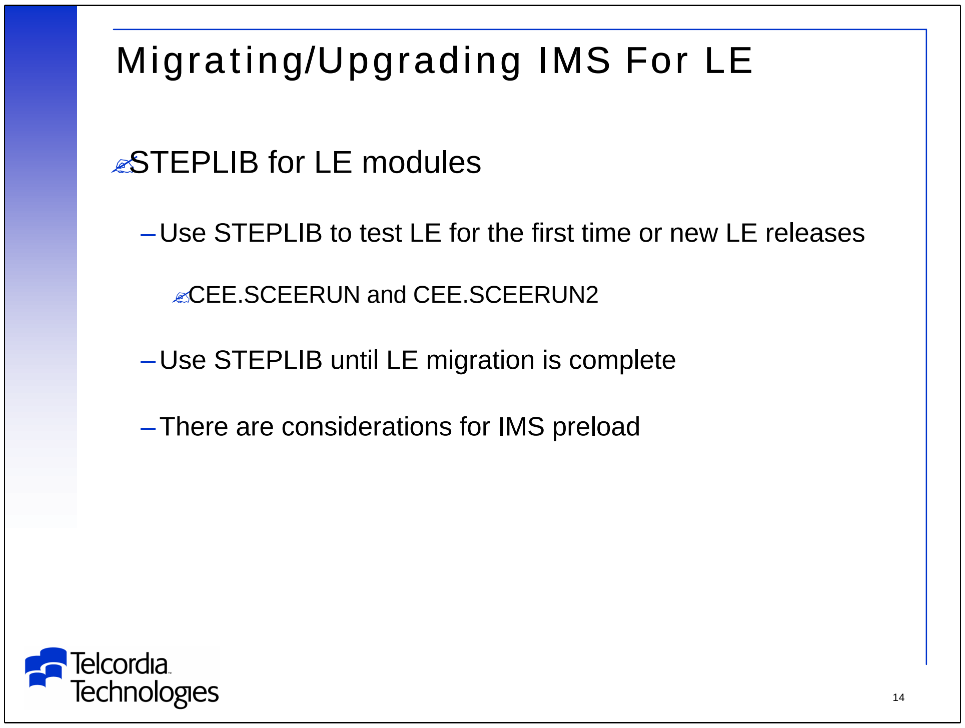
13

LE runtime libraries SCEERUN and SCEERUN2 can be placed in LNKLIST.
-SCEERUN2 is new for OS/390 R2 and higher releases.
-This library currently contains only XPLINK.

LPA reduces overall system storage use since routines are sharable.

LPA reduces initialization and termination time since load time decreases.

II10425 is old and for OS/390.
-Installing LE not using LNKLIST is harder to do for z/OS.



Migrating/Upgrading IMS For LE

- STEPLIB for LE modules
  - Use STEPLIB to test LE for the first time or new LE releases
    - CEE.SCEERUN and CEE.SCEERUN2
  - Use STEPLIB until LE migration is complete
  - There are considerations for IMS preload

Telcordia Technologies

14

TEST – Put CEE.SCEERUN (and CEE.SCEERUN2) into STEPLIB and regression test.

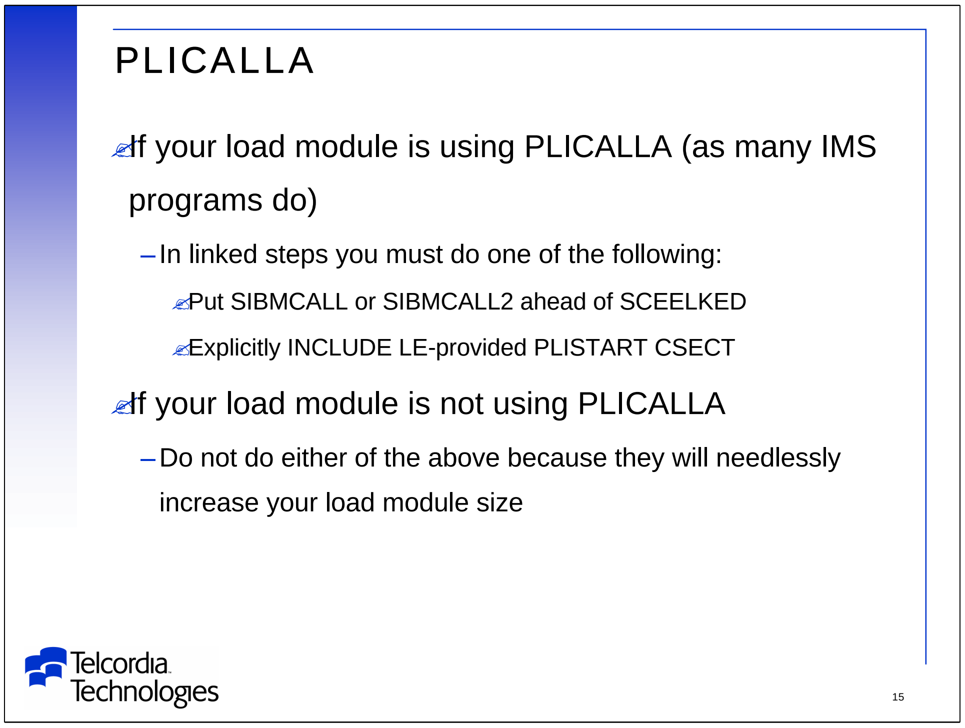
-Do not forget to test error scripts and On Error conditions.

PRELOAD – When using STEPLIB for LE libraries in IMS Dependent Regions any LE library routines that are preloaded are not loaded into read-only storage.

-If your application has an error it can overwrite the preloaded routines causing corruption and abends.

-If there is an abend preloaded reentrant modules are not reloaded unless the entire dependent region is recycled so the abends may be persistent.

-We have developed a user mod which loads the modules into protected storage and catches violators with an 0C4 abend.



PLICALLA

- If your load module is using PLICALLA (as many IMS programs do)
  - In linked steps you must do one of the following:
    - Put SIBMCALL or SIBMCALL2 ahead of SCEELKED
    - Explicitly INCLUDE LE-provided PLISTART CSECT
- If your load module is not using PLICALLA
  - Do not do either of the above because they will needlessly increase your load module size

Telcordia Technologies

15

LE provided support for OS PL/I applications that use the PLICALLA entry point (but not without a fight).

-LE also provides for recompiled OS PL/I applications that want to continue to use PLICALLA as the primary entry point

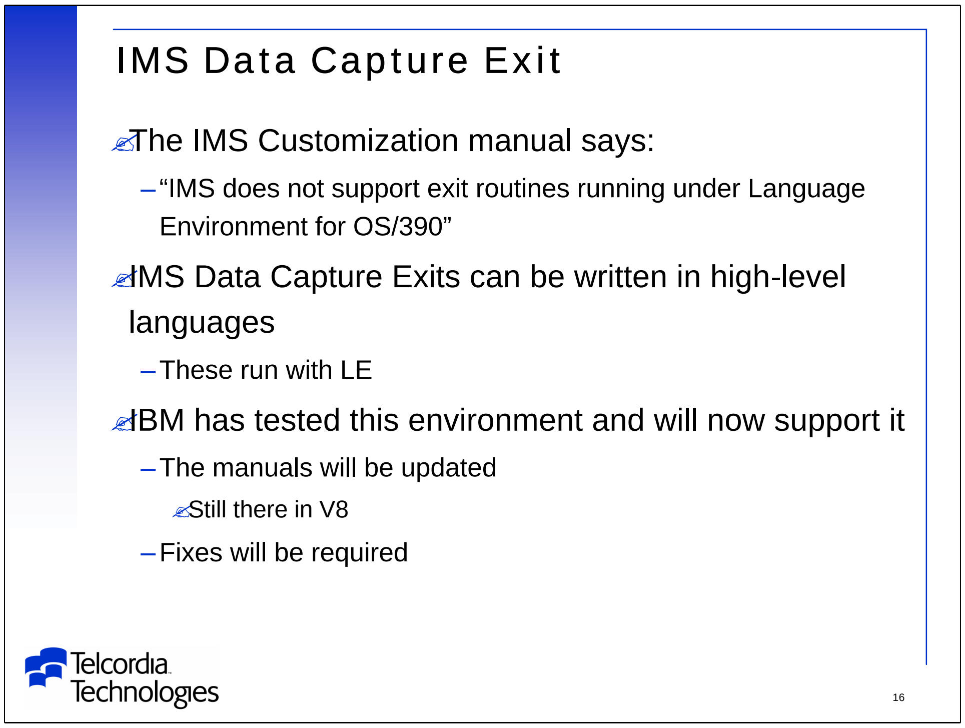
If you recompile an OS PL/I program with PLICALLA with PL/I for MVS & VM you must do one of the following when you linkedit your main load module:

-For MVS applications concatenate SIBMCALL before SCEELKED

-For VM applications global SIBMCALL before SCEELKED

-Explicitly INLUDE the LE-provided PLISTART CSECT

If you do not do this the linkage editor or loader will issue an error message for an unresolved ENTRY PLICALLA statement.



IMS Data Capture Exit

- The IMS Customization manual says:
  - “IMS does not support exit routines running under Language Environment for OS/390”
- IMS Data Capture Exits can be written in high-level languages
  - These run with LE
- IBM has tested this environment and will now support it
  - The manuals will be updated
    - Still there in V8
  - Fixes will be required

Telcordia Technologies

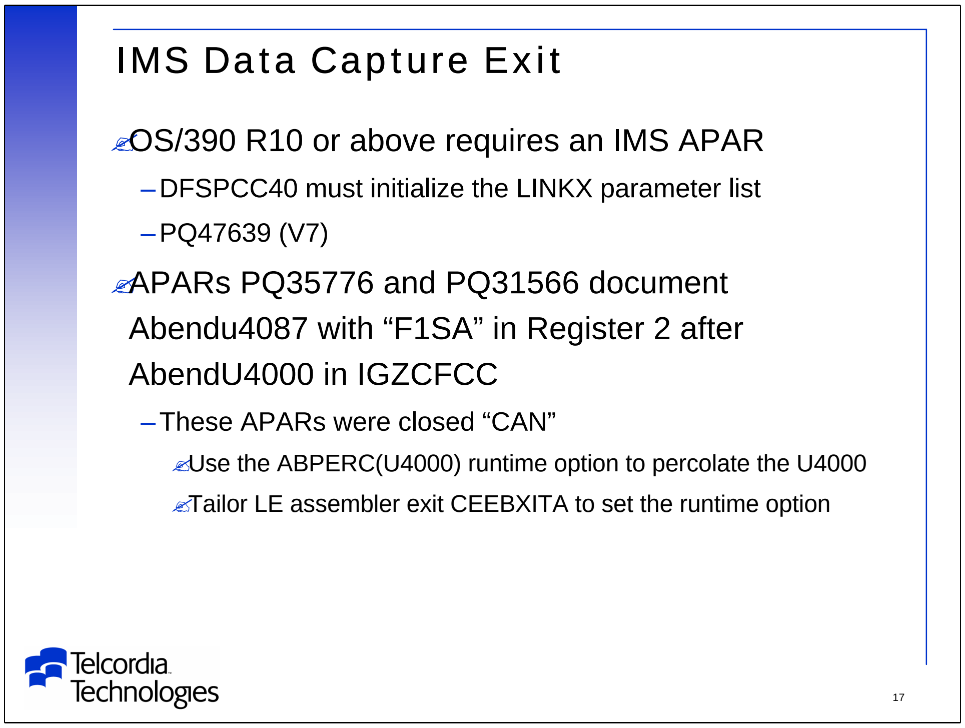
16

There were problems running IMS Data Capture Exits written in high-level languages (PL/I for one).

-These are documented on the next foil.

The IMS V8 documentation will be updated to say that this environment is now supported.

- But not in the level of V8 manuals on my CD.



IMS Data Capture Exit

- OS/390 R10 or above requires an IMS APAR
  - DFSPCC40 must initialize the LINKX parameter list
  - PQ47639 (V7)
- APARs PQ35776 and PQ31566 document Abendu4087 with “F1SA” in Register 2 after AbendU4000 in IGZCFCC
  - These APARs were closed “CAN”
    - Use the ABPERC(U4000) runtime option to percolate the U4000
    - Tailor LE assembler exit CEEBXITA to set the runtime option

Telcordia Technologies

17

IMS uses LINKX to call the Data Capture Exit.

-APAR PQ46980 corrects abends caused when IMS passes a ‘dirty’ parameter list to LINKX and OS/390 R10 tests a previously reserved bit.

-These abends can be U4087, U4039, U4036, etc.

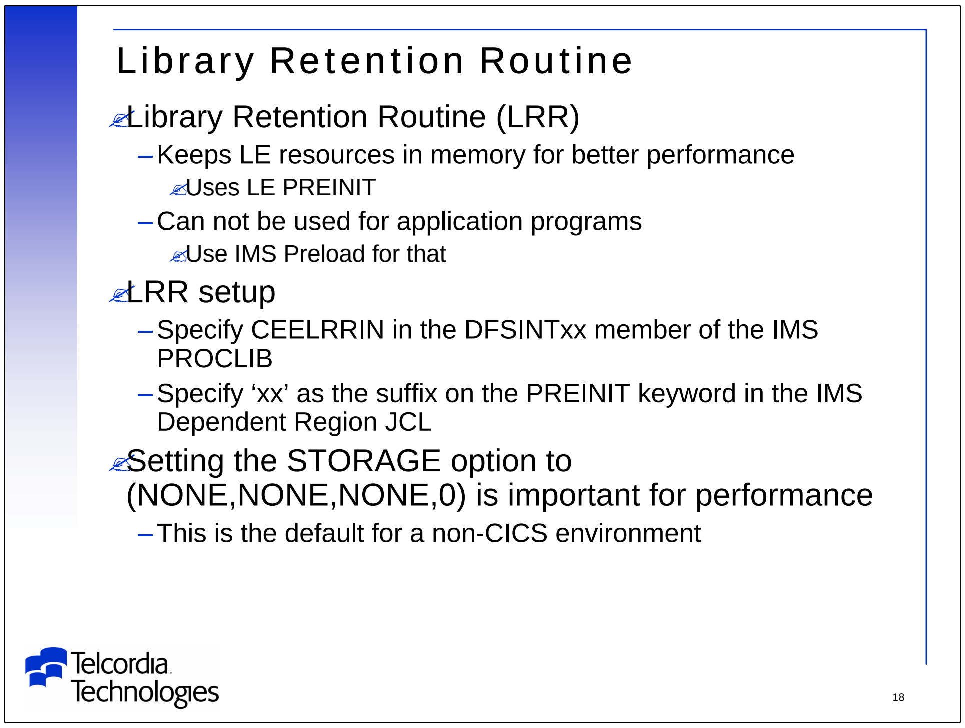
-The APAR mentions COBOL but this can happen with exits written in other languages.

-We hit it with PL/I

APARs PQ35766 and PQ31566 document a problem where LE tried to verify the stack (save area chain) and found an indicator for ‘linkage stack’

-LE does not support the use of linkage stacks

-The solution is to percolate the original abend and avoid the stack verification



Library Retention Routine

- Library Retention Routine (LRR)
  - Keeps LE resources in memory for better performance
    - Uses LE PREINIT
  - Can not be used for application programs
    - Use IMS Preload for that
- LRR setup
  - Specify CEELRRIN in the DFSINTxx member of the IMS PROCLIB
  - Specify ‘xx’ as the suffix on the PREINIT keyword in the IMS Dependent Region JCL
- Setting the STORAGE option to (NONE,NONE,NONE,0) is important for performance
  - This is the default for a non-CICS environment

Telcordia Technologies

18

If you are running LE in an IMS dependent region you can improve performance using LRR.

The following is a partial list of resources which LE keeps in memory with LRR in effect:

-LE runtime load modules

-LE storage associated with the management of the runtime load modules

-LE storage for start-up control blocks

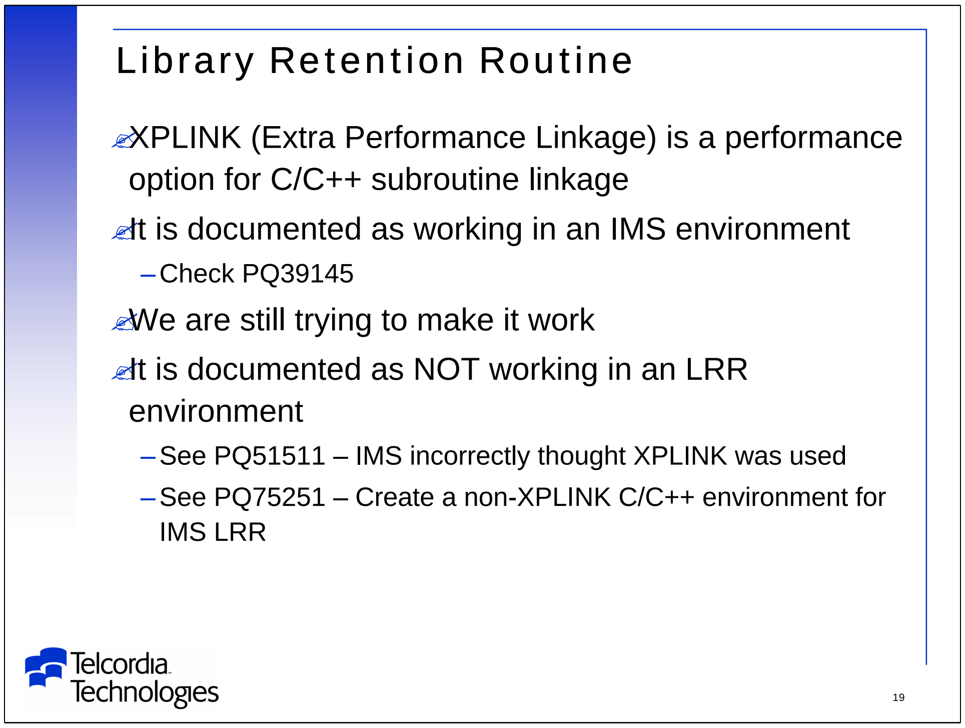
LRR does not manage application programs.

-Use IMS Preload

LRR does NOT work with applications using LE PICI (Program Interrupt Callable Interface).

- IMS may never support this environment.

See the LE Customization manual for setup information.



Library Retention Routine

- XPLINK (Extra Performance Linkage) is a performance option for C/C++ subroutine linkage
- It is documented as working in an IMS environment
  - Check PQ39145
- We are still trying to make it work
- It is documented as NOT working in an LRR environment
  - See PQ51511 – IMS incorrectly thought XPLINK was used
  - See PQ75251 – Create a non-XPLINK C/C++ environment for IMS LRR

Telcordia Technologies

19

XPLINK is a high-performance compile time option for C/C++ programs.

It is used to improve performance of C/C++ subroutine calls.

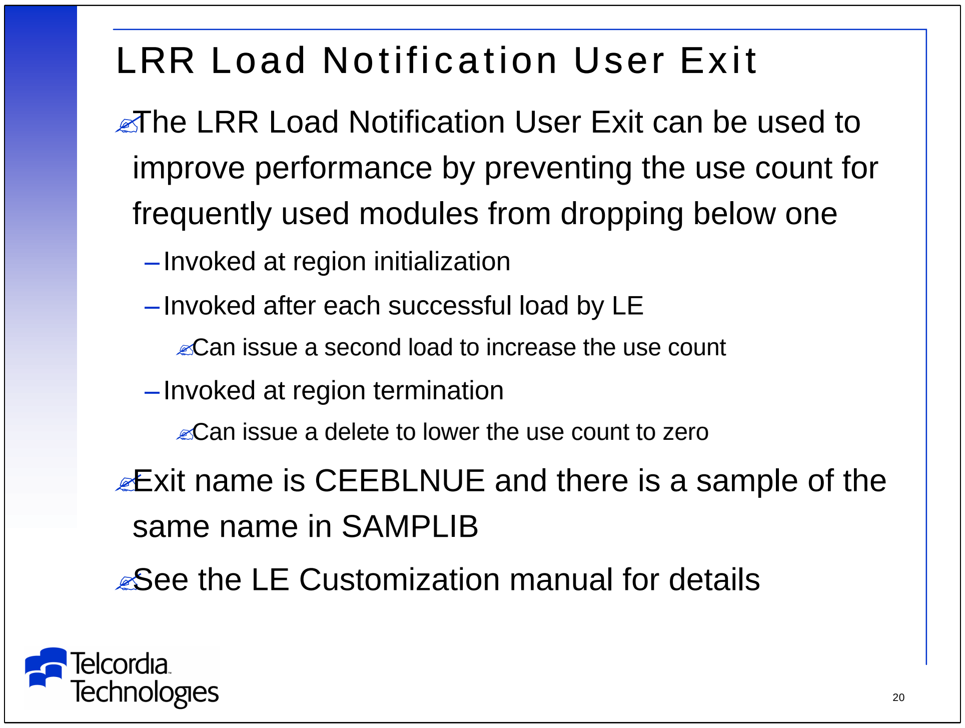
It is supposed to work in an IMS environment without LRR but it does not.

We are currently working with IBM to fix this.

It would seem to me to be useless if it does not also work with LRR.

PQ75251 creates a non-XPLINK Standard C++ Library DLL to run under IMS LRR.

PQ59972 documents an 0C1 in CEETDLI using IPALINK.



LRR Load Notification User Exit

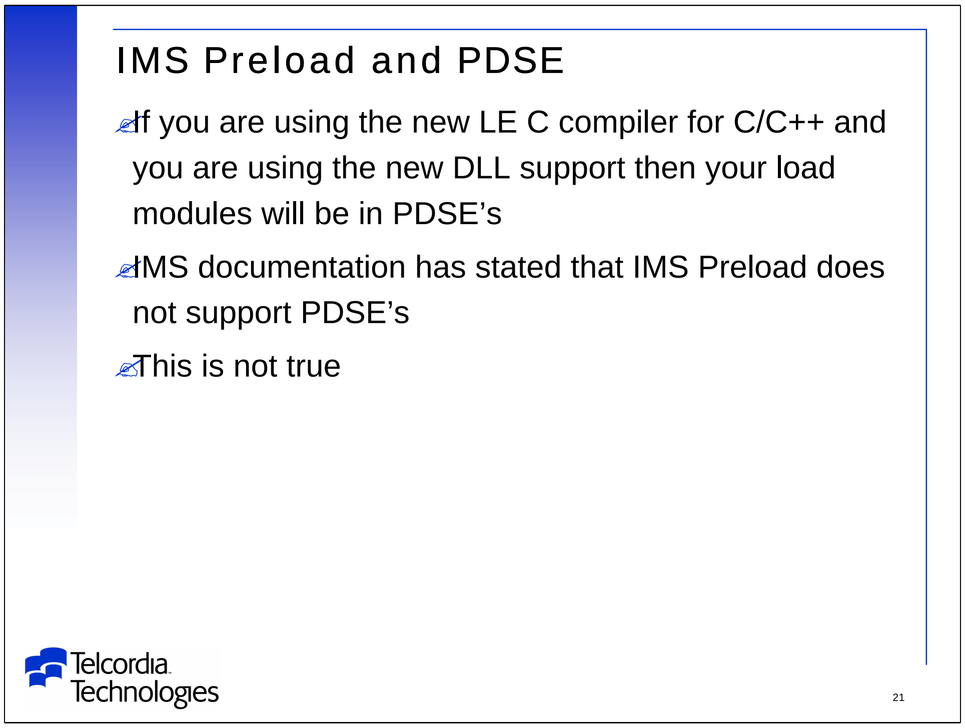
- The LRR Load Notification User Exit can be used to improve performance by preventing the use count for frequently used modules from dropping below one
  - Invoked at region initialization
  - Invoked after each successful load by LE
    - Can issue a second load to increase the use count
  - Invoked at region termination
    - Can issue a delete to lower the use count to zero
- Exit name is CEEBLNUE and there is a sample of the same name in SAMPLIB
- See the LE Customization manual for details

Telcordia Technologies

20

Use of the LRR Load Notification User Exit requires Library Retention Routine (LRR) to be active.

We have not tested this.



IMS Preload and PDSE

- If you are using the new LE C compiler for C/C++ and you are using the new DLL support then your load modules will be in PDSE's
- IMS documentation has stated that IMS Preload does not support PDSE's
- This is not true

Telcordia Technologies

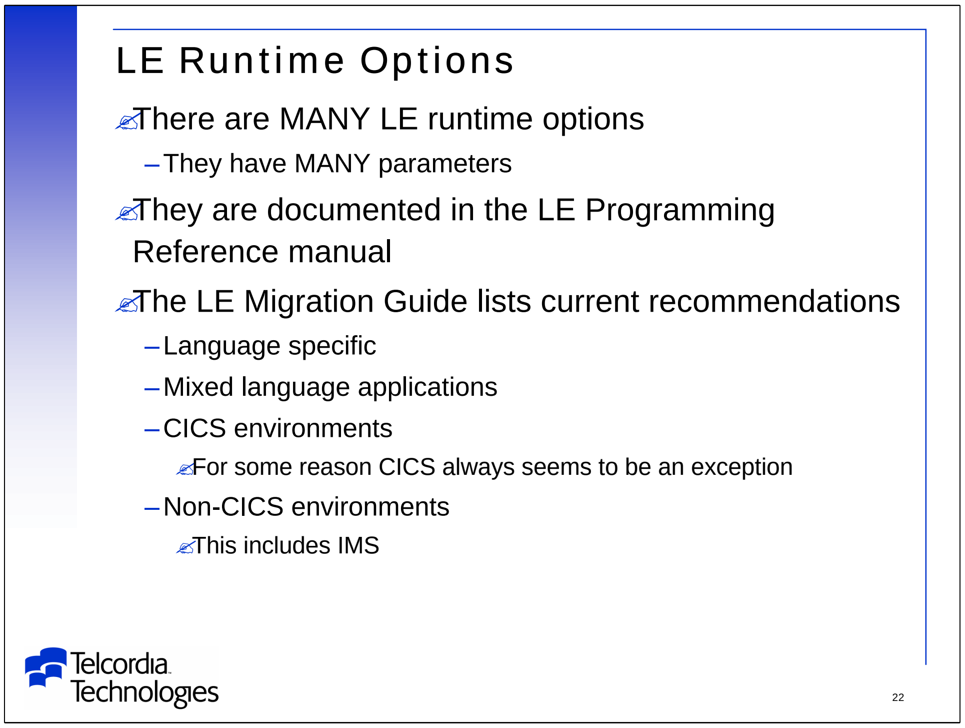
21

There were IMS informational APAR's that stated that IMS did not support preloading modules which were in PDSE's.

This is not true.

It works fine for us.

IMS Level 2 (John Butterweck) did extensive research on this and determined that preloading from PDSE's is supported.



LE Runtime Options

- There are MANY LE runtime options
  - They have MANY parameters
- They are documented in the LE Programming Reference manual
- The LE Migration Guide lists current recommendations
  - Language specific
  - Mixed language applications
  - CICS environments
    - For some reason CICS always seems to be an exception
  - Non-CICS environments
    - This includes IMS

Telcordia Technologies

22

There are many LE runtime options and these options have many parameters.

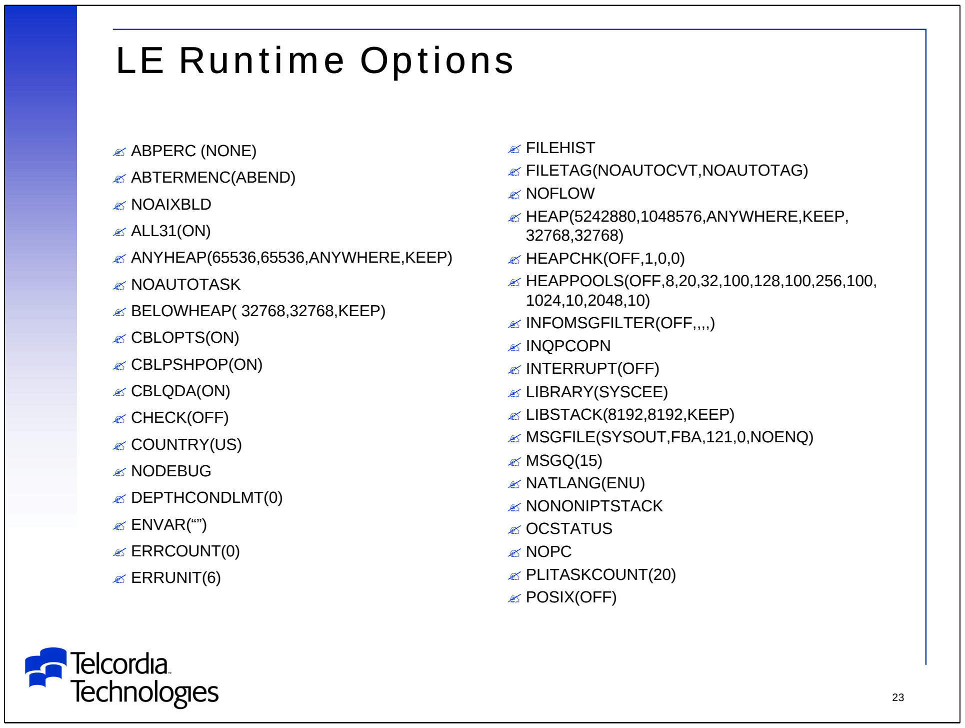
There are many ways of setting these runtime options.

This presentation will cover those that have been found to influence IMS execution and performance.

Question – how to you spell “runtime”?

-LE uses “Run-Time” in the titles of the manuals.

-LE uses “runtime” in the text of the manuals.



LE Runtime Options

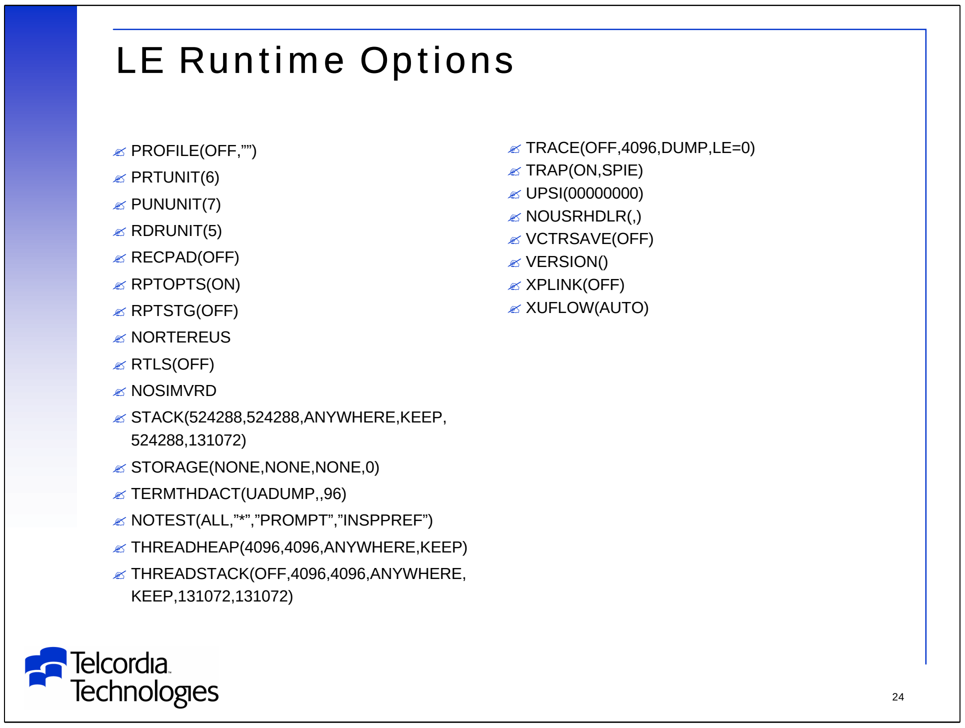
- ABPERC (NONE)
- ABTERMENC(ABEND)
- NOAIXBLD
- ALL31(ON)
- ANYHEAP(65536,65536,ANYWHERE,KEEP)
- NOAUTOTASK
- BELOWHEAP(32768,32768,KEEP)
- CBLOPTS(ON)
- CBLPSHPOP(ON)
- CBLQDA(ON)
- CHECK(OFF)
- COUNTRY(US)
- NODEBUG
- DEPTHCONDLMT(0)
- ENVAR(“”)
- ERRCOUNT(0)
- ERRUNIT(6)
- FILEHIST
- FILETAG(NOAUTOCVT,NOAUTOTAG)
- NOFLOW
- HEAP(5242880,1048576,ANYWHERE,KEEP, 32768,32768)
- HEAPCHK(OFF,1,0,0)
- HEAPPOOLS(OFF,8,20,32,100,128,100,256,100, 1024,10,2048,10)
- INFOMSGFILTER(OFF,,,,)
- INQPCOPN
- INTERRUPT(OFF)
- LIBRARY(SYSCEE)
- LIBSTACK(8192,8192,KEEP)
- MSGFILE(SYSOUT,FBA,121,0,NOENQ)
- MSGQ(15)
- NATLANG(ENU)
- NONONIPTSTACK
- OCSTATUS
- NOPC
- PLITASKCOUNT(20)
- POSIX(OFF)

Telcordia Technologies

23

There are many LE runtime options and parameters.

Here is a list of a RPTOPTS output for one application.



LE Runtime Options

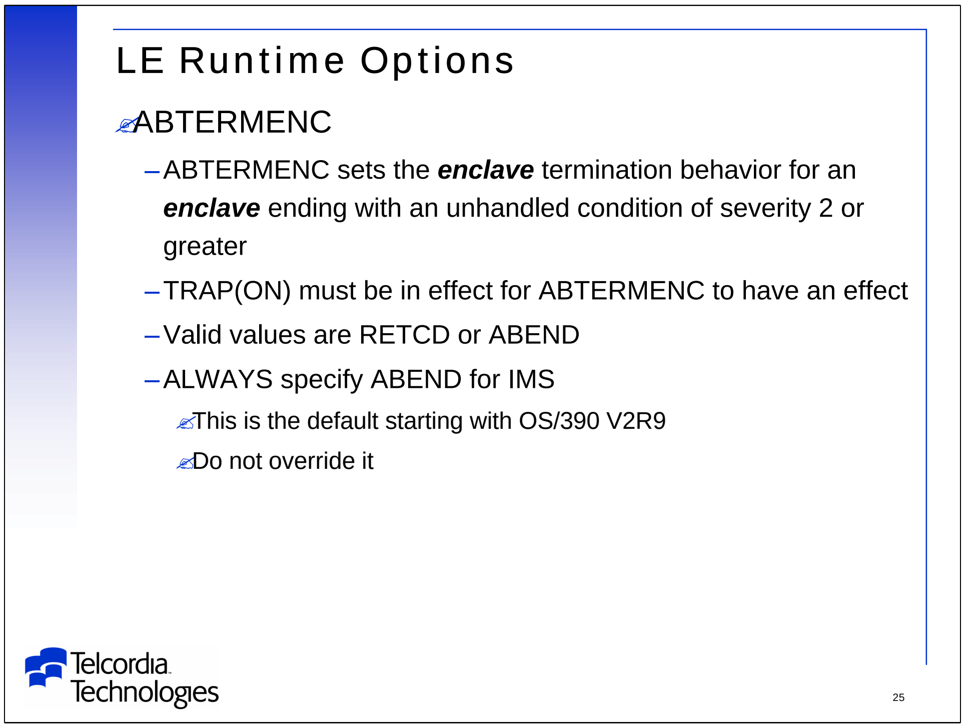
- PROFILE(OFF,"")
- PRTUNIT(6)
- PUNUNIT(7)
- RDRUNIT(5)
- RECPAD(OFF)
- RPTOPTS(ON)
- RPTSTG(OFF)
- NORTEREUS
- RTLS(OFF)
- NOSIMVRD
- STACK(524288,524288,ANYWHERE,KEEP, 524288,131072)
- STORAGE(NONE,NONE,NONE,0)
- TERMTHDACT(UADUMP,,96)
- NOTEST(ALL,"*","PROMPT","INSPPREF")
- THREADHEAP(4096,4096,ANYWHERE,KEEP)
- THREADSTACK(OFF,4096,4096,ANYWHERE, KEEP,131072,131072)
- TRACE(OFF,4096,DUMP,LE=0)
- TRAP(ON,SPIE)
- UPSI(00000000)
- NOUSRHDLR(,)
- VCTRSAVE(OFF)
- VERSION()
- XPLINK(OFF)
- XUFLOW(AUTO)

Telcordia Technologies

24

There are many LE runtime options and parameters.

Here is the rest of the list of a RPTOPTS output for one application.



LE Runtime Options

- ABTERMENC
  - ABTERMENC sets the ***enclave*** termination behavior for an ***enclave*** ending with an unhandled condition of severity 2 or greater
  - TRAP(ON) must be in effect for ABTERMENC to have an effect
  - Valid values are RETCD or ABEND
  - ALWAYS specify ABEND for IMS
    - This is the default starting with OS/390 V2R9
    - Do not override it

Telcordia Technologies

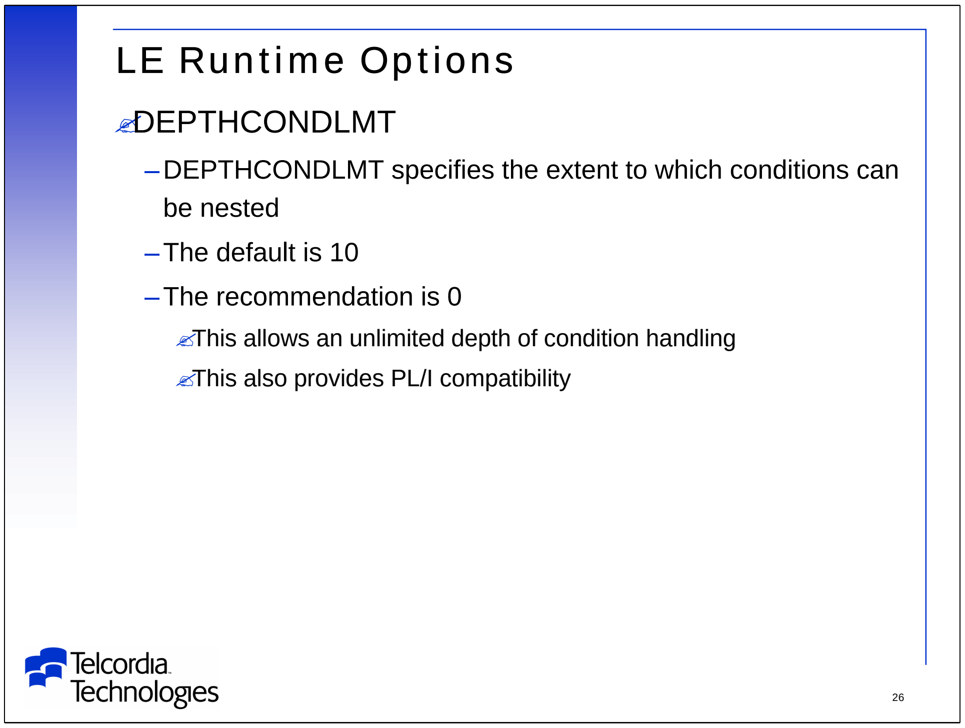
25

ABTERMENC(ABEND) is the default for V2R9 and above.

-Earlier releases had RETCODE as the default.

Specifying RETCODE can cause problems with rollback under IMS because errors in application programs result in getting RC=3000 rather than

an abend and IMS and DB2 (and MQSeries) do not know that backout is required.



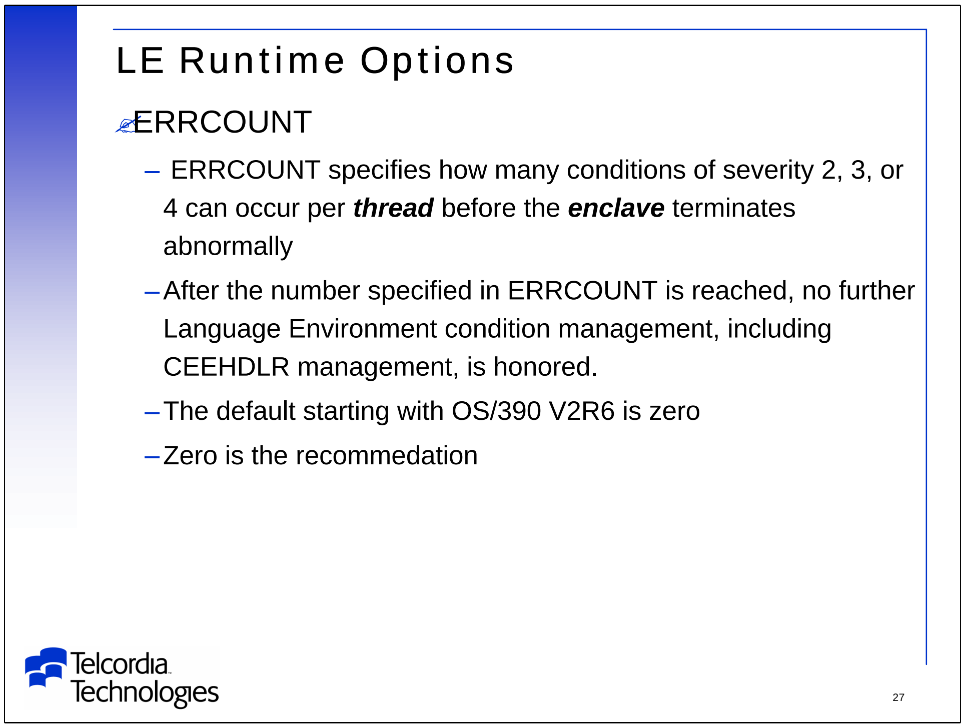
LE Runtime Options

- DEPTHCONDLMT
  - DEPTHCONDLMT specifies the extent to which conditions can be nested
  - The default is 10
  - The recommendation is 0
    - This allows an unlimited depth of condition handling
    - This also provides PL/I compatibility

Telcordia Technologies

26

If the limit is not 0 and the limit is reached LE will not be able to handle the error.



LE Runtime Options

- ERRCOUNT
  - ERRCOUNT specifies how many conditions of severity 2, 3, or 4 can occur per ***thread*** before the ***enclave*** terminates abnormally
  - After the number specified in ERRCOUNT is reached, no further Language Environment condition management, including CEEHDLR management, is honored.
  - The default starting with OS/390 V2R6 is zero
  - Zero is the recommedation

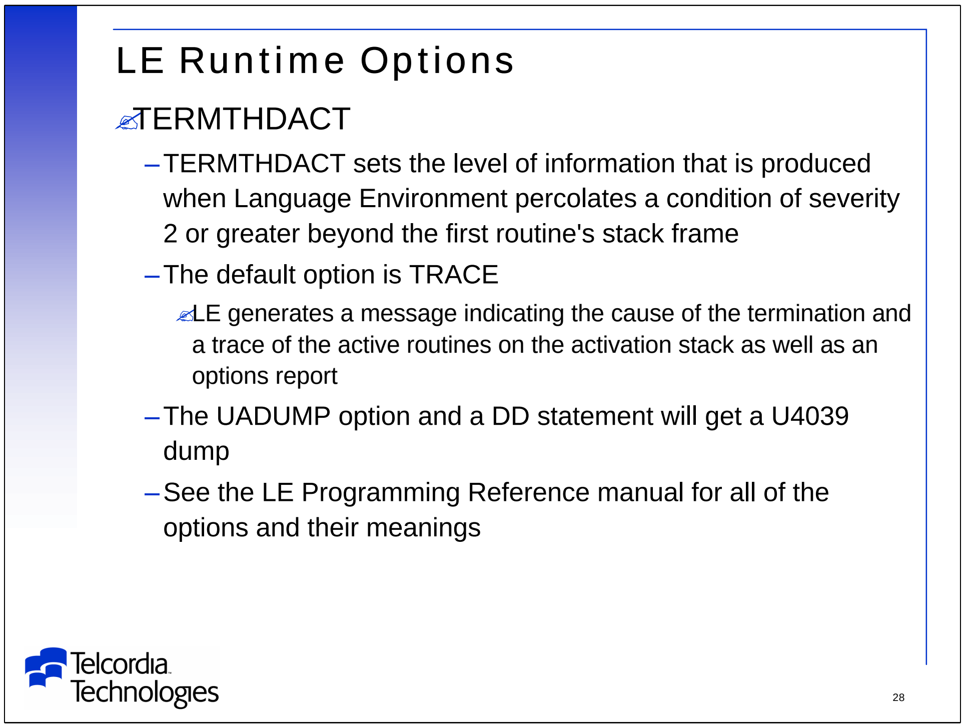
Telcordia Technologies

27

A value of zero means that the LE condition handler will not terminate the task regardless of the number of conditions that are generated.

It can be normal for some conditions, such as PL/I ENDPAGE to occur many times in an application.

Setting ERRCOUNT(0) can avoid unnecessary abends.



LE Runtime Options

- TERMTHDACT
  - TERMTHDACT sets the level of information that is produced when Language Environment percolates a condition of severity 2 or greater beyond the first routine's stack frame
  - The default option is TRACE
    - LE generates a message indicating the cause of the termination and a trace of the active routines on the activation stack as well as an options report
  - The UADUMP option and a DD statement will get a U4039 dump
  - See the LE Programming Reference manual for all of the options and their meanings

Telcordia Technologies

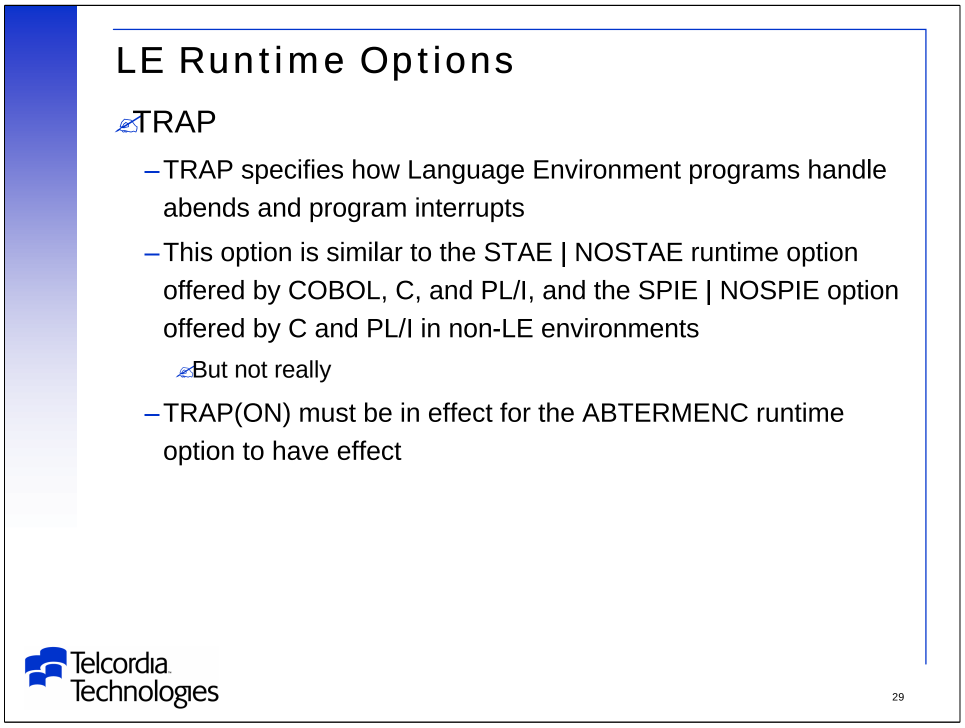
28

If you are used to REAL MVS dumps LE snaps can be very ugly and not very useful.

There are many options described in the LE Programming Reference manual.

This has no affect on how LE notifies IMS when there is an abend.

PQ58958 (+ PQ66066) fixes a problem when there is a U474 abend (/STO REGION x ABDUMP) and no dump is produced.



LE Runtime Options

- TRAP
  - TRAP specifies how Language Environment programs handle abends and program interrupts
  - This option is similar to the STAE | NOSTAE runtime option offered by COBOL, C, and PL/I, and the SPIE | NOSPIE option offered by C and PL/I in non-LE environments
    - But not really
  - TRAP(ON) must be in effect for the ABTERMENC runtime option to have effect

Telcordia Technologies

29

TRAP (ON,SPIE) fully enables the LE condition handler.

Due to restrictions and side-effects when running TRAP(OFF) IBM highly recommends running TRAP(ON) in all environments – including IMS.

- LE will pass the abend to IMS for proper error handling.

TRAP(OFF) does not totally nullify the LE condition handler – there is still an ESTAE.

-There may be cases, e.g. traversing the DSA chain, where an 0C4 can be trapped and ignored.

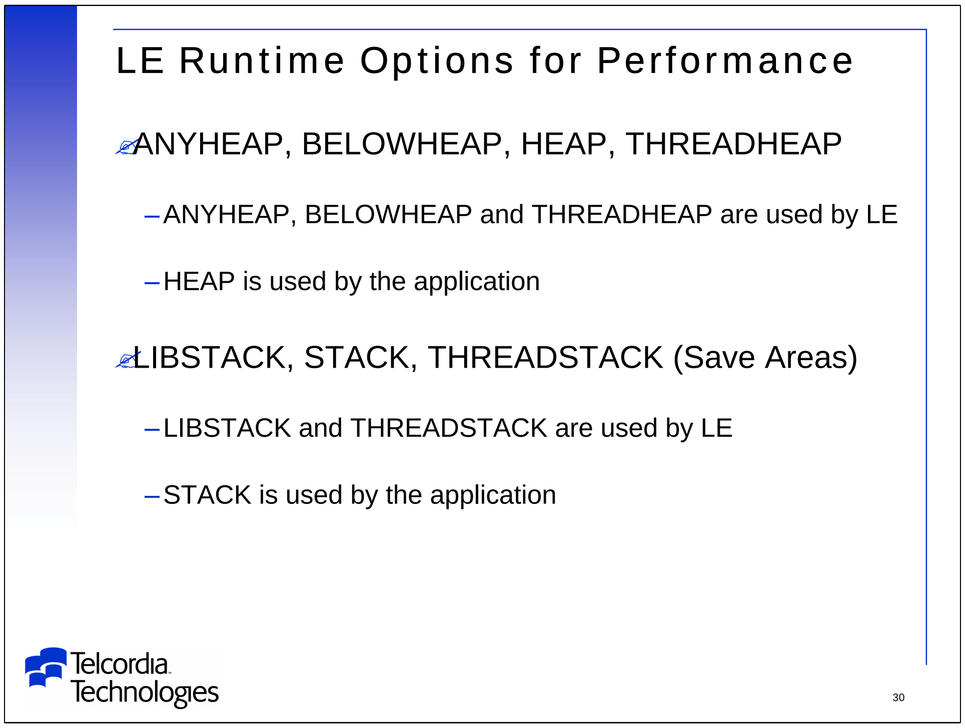
-This is known as the shunting mechanism.

If you really want to control the error handling there is an LE API called CEE3ERP.

-This was created for Telcordia.

-It allows a local error handler to be invoked in a shunting situation to determine the abend action.

-This is documented in the LE Vendor Interface manual.



LE Runtime Options for Performance

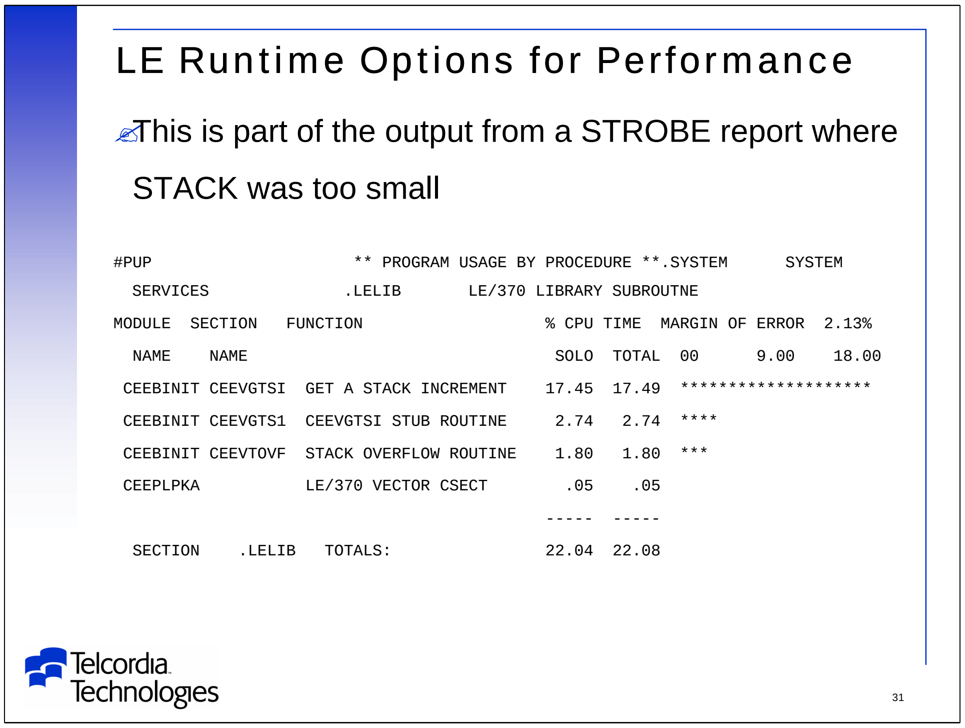
- ANYHEAP, BELOWHEAP, HEAP, THREADHEAP
  - ANYHEAP, BELOWHEAP and THREADHEAP are used by LE
  - HEAP is used by the application
- LIBSTACK, STACK, THREADSTACK (Save Areas)
  - LIBSTACK and THREADSTACK are used by LE
  - STACK is used by the application

Telcordia Technologies

30

These options improve performance when you specify values that minimize the number of times the operating system allocates storage

-And can absolutely kill performance if they are too small



LE Runtime Options for Performance

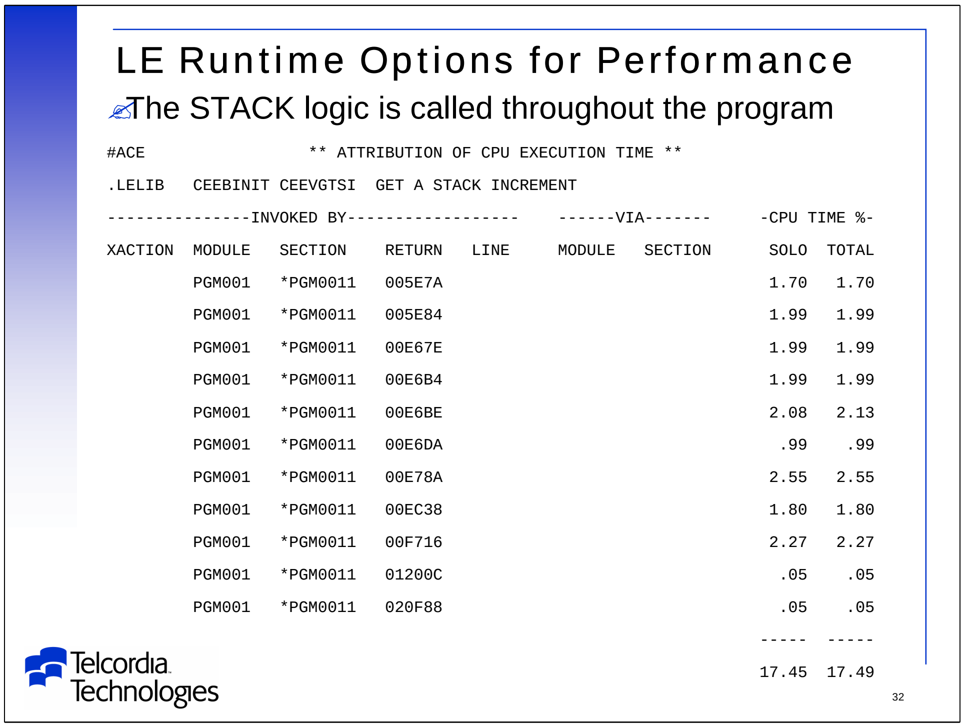
- This is part of the output from a STROBE report where STACK was too small

```
#PUP                     ** PROGRAM USAGE BY PROCEDURE **.SYSTEM      SYSTEM
  SERVICES              .LELIB       LE/370 LIBRARY SUBROUTNE
MODULE  SECTION   FUNCTION                    % CPU TIME  MARGIN OF ERROR  2.13%
  NAME    NAME                                 SOLO  TOTAL  00      9.00    18.00
 CEEBINIT CEEVGTSI  GET A STACK INCREMENT     17.45  17.49  ********************
 CEEBINIT CEEVGTS1  CEEVGTSI STUB ROUTINE      2.74   2.74  ****
 CEEBINIT CEEVTOVF  STACK OVERFLOW ROUTINE     1.80   1.80  ***
 CEEPLPKA           LE/370 VECTOR CSECT         .05    .05
                                              -----  -----
  SECTION    .LELIB   TOTALS:                 22.04  22.08
```

Telcordia Technologies

31

In this case 22% of the CPU was wasted because the STACK value was too small.



LE Runtime Options for Performance

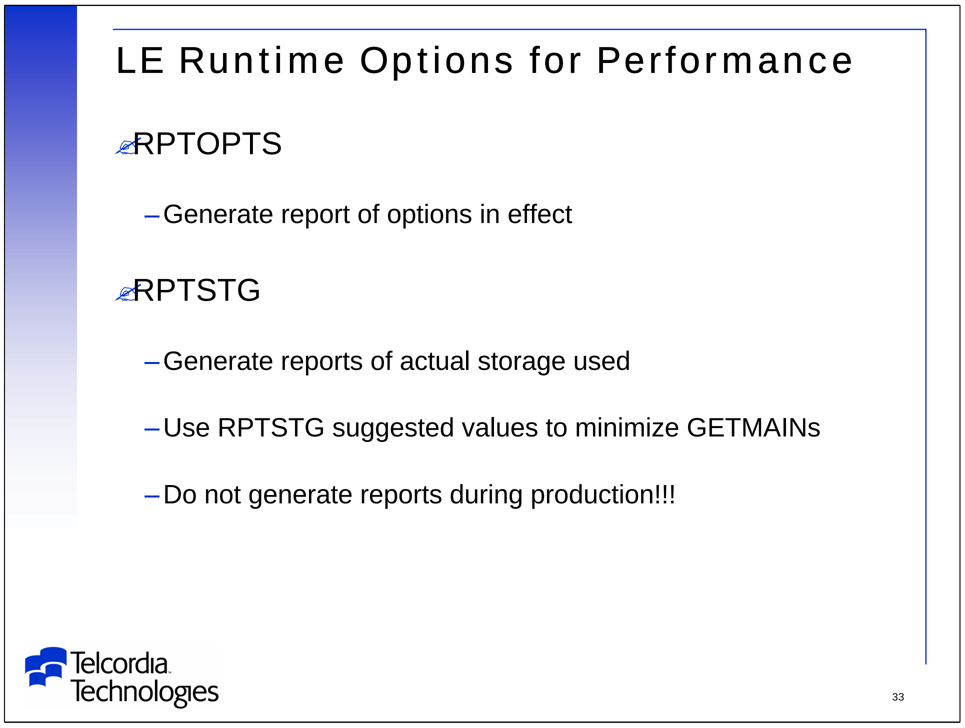
The STACK logic is called throughout the program

```
#ACE                 ** ATTRIBUTION OF CPU EXECUTION TIME **

.LELIB   CEEBINIT CEEVGTSI  GET A STACK INCREMENT

---------------INVOKED BY------------------    ------VIA-------     -CPU TIME %-

XACTION  MODULE   SECTION    RETURN   LINE     MODULE   SECTION       SOLO  TOTAL

         PGM001   *PGM0011   005E7A                                   1.70   1.70

         PGM001   *PGM0011   005E84                                   1.99   1.99

         PGM001   *PGM0011   00E67E                                   1.99   1.99

         PGM001   *PGM0011   00E6B4                                   1.99   1.99

         PGM001   *PGM0011   00E6BE                                   2.08   2.13

         PGM001   *PGM0011   00E6DA                                    .99    .99

         PGM001   *PGM0011   00E78A                                   2.55   2.55

         PGM001   *PGM0011   00EC38                                   1.80   1.80

         PGM001   *PGM0011   00F716                                   2.27   2.27

         PGM001   *PGM0011   01200C                                    .05    .05

         PGM001   *PGM0011   020F88                                    .05    .05

                                                                     -----  -----

                                                                     17.45  17.49
```

Telcordia Technologies

32

In this case 22% of the CPU was wasted because the STACK value was too small.



LE Runtime Options for Performance

- RPTOPTS
  - Generate report of options in effect
- RPTSTG
  - Generate reports of actual storage used
  - Use RPTSTG suggested values to minimize GETMAINs
  - Do not generate reports during production!!!

Telcordia Technologies

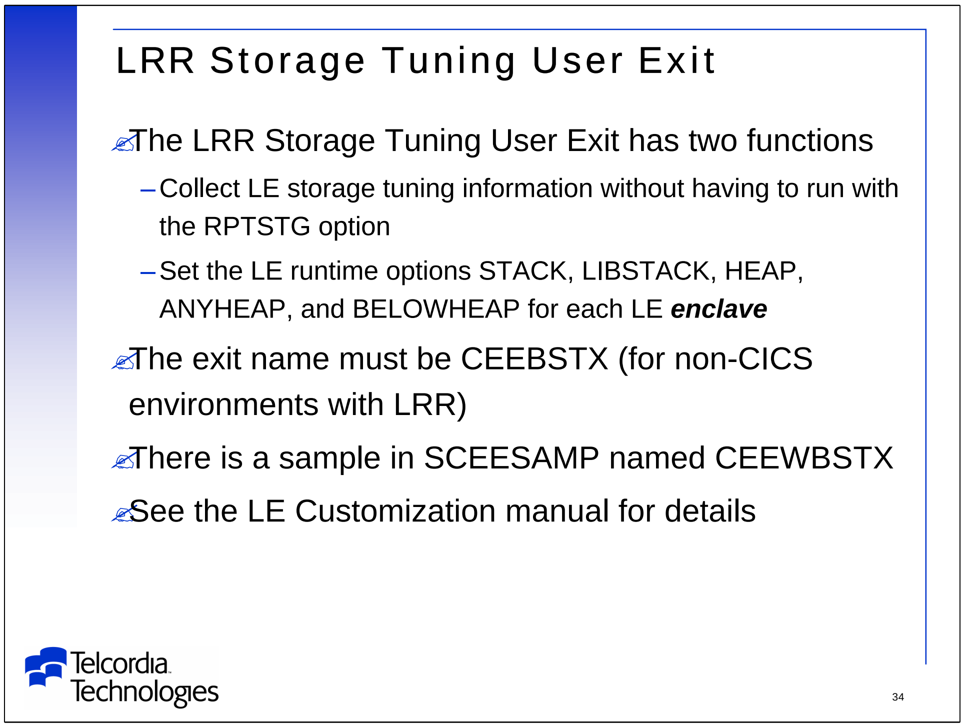
33

The RPTSTG runtime option generates a report of the storage the application uses while it is running

-You can use the report to establish values for these parameters

The RPTOPTS and RPTSTG options increase the time it takes for the application to execute

-Use them only as an aid to application development



LRR Storage Tuning User Exit

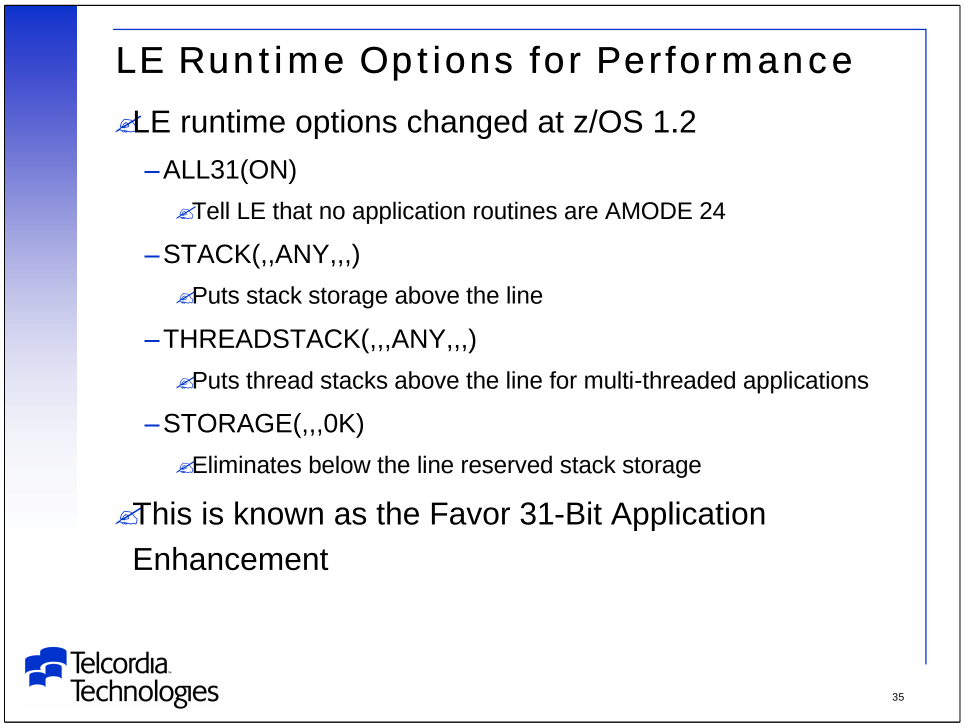
- The LRR Storage Tuning User Exit has two functions
  - Collect LE storage tuning information without having to run with the RPTSTG option
  - Set the LE runtime options STACK, LIBSTACK, HEAP, ANYHEAP, and BELOWHEAP for each LE ***enclave***
- The exit name must be CEEBSTX (for non-CICS environments with LRR)
- There is a sample in SCEESAMP named CEEWBSTX
- See the LE Customization manual for details

Telcordia Technologies

34

This looks like a great feature.

-We have not tested it.



LE Runtime Options for Performance

- LE runtime options changed at z/OS 1.2
  - ALL31(ON)
    - Tell LE that no application routines are AMODE 24
  - STACK(,,ANY,,,)
    - Puts stack storage above the line
  - THREADSTACK(,,,ANY,,,)
    - Puts thread stacks above the line for multi-threaded applications
  - STORAGE(,,,0K)
    - Eliminates below the line reserved stack storage
- This is known as the Favor 31-Bit Application Enhancement

Telcordia Technologies

35

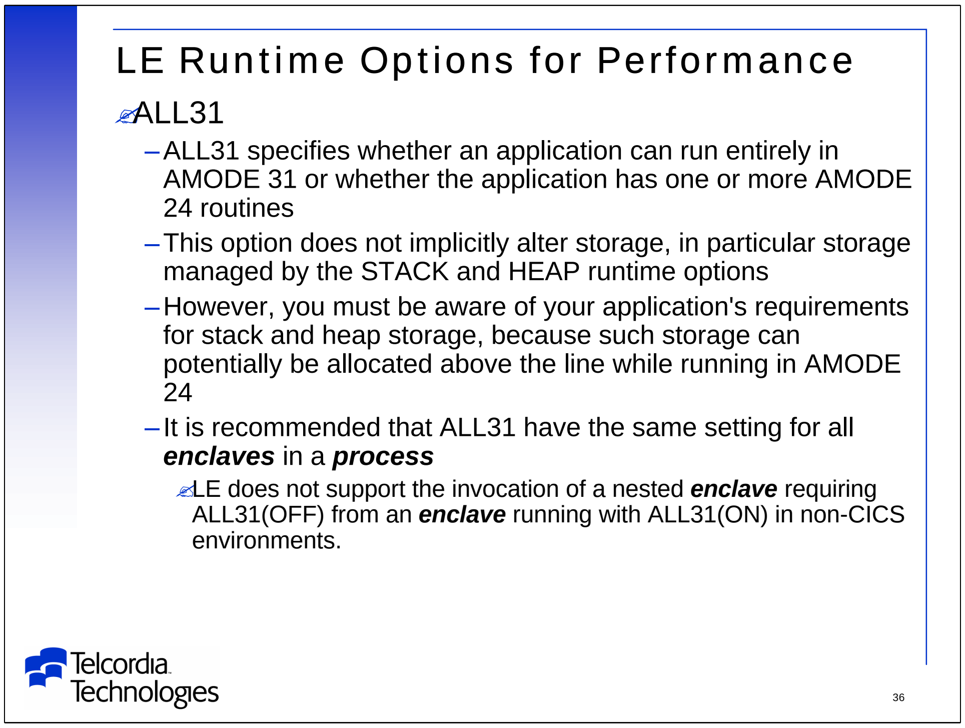
Z/OS 1.2 LE includes an enhancement to favor 31-bit applications.

-The goals of this enhancement are to reduce init/term path length and reduce below-the-line storage usage.

-These changes introduce migration concerns for non-CICS applications that invoke AMODE 24 routines.

The reserve stack is used to process an out-of-storage condition, as when a stack or heap can not be extended.

The LE Runtime Migration manual has details of these changes if you have AMODE 24 routines.


LE Runtime Options for Performance

- ALL31
  - ALL31 specifies whether an application can run entirely in AMODE 31 or whether the application has one or more AMODE 24 routines
  - This option does not implicitly alter storage, in particular storage managed by the STACK and HEAP runtime options
  - However, you must be aware of your application's requirements for stack and heap storage, because such storage can potentially be allocated above the line while running in AMODE 24
  - It is recommended that ALL31 have the same setting for all ***enclaves*** in a ***process***
    - LE does not support the invocation of a nested ***enclave*** requiring ALL31(OFF) from an ***enclave*** running with ALL31(ON) in non-CICS environments.

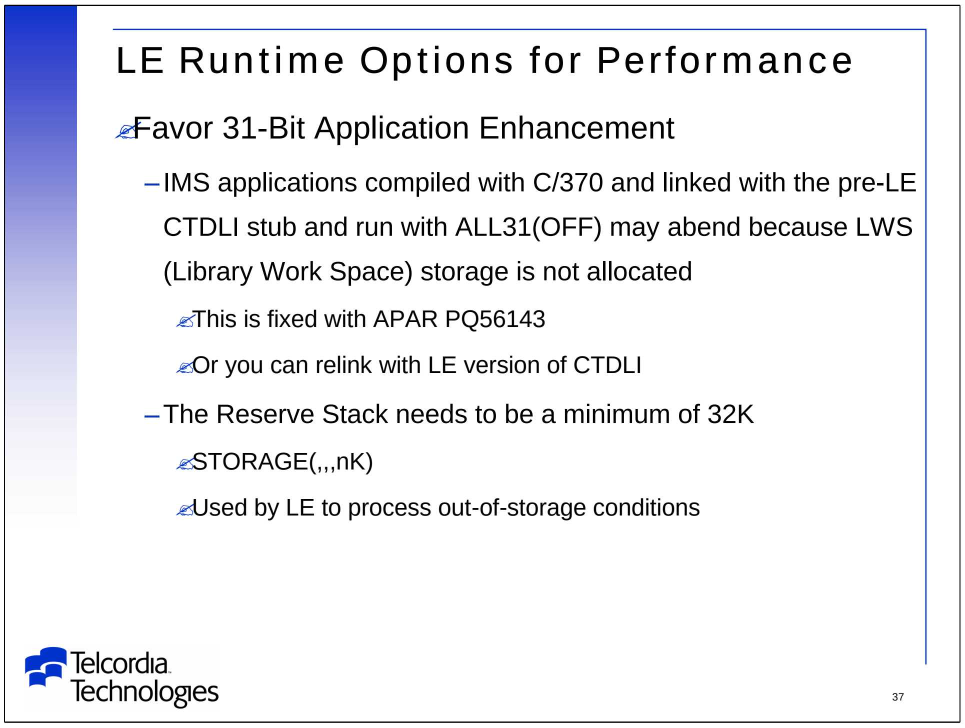
Telcordia Technologies

36

If your application consists entirely of AMODE 31 routines it wll run faster and use less below-the-line storage with ALL31(ON) than with ALL31(OFF).

If ALL31(ON) is in effect there is no AMODE switching among library routines.

If you still have AMODE 24 programs, they must use ALL31(OFF) and STACK(,,BELOW).



LE Runtime Options for Performance

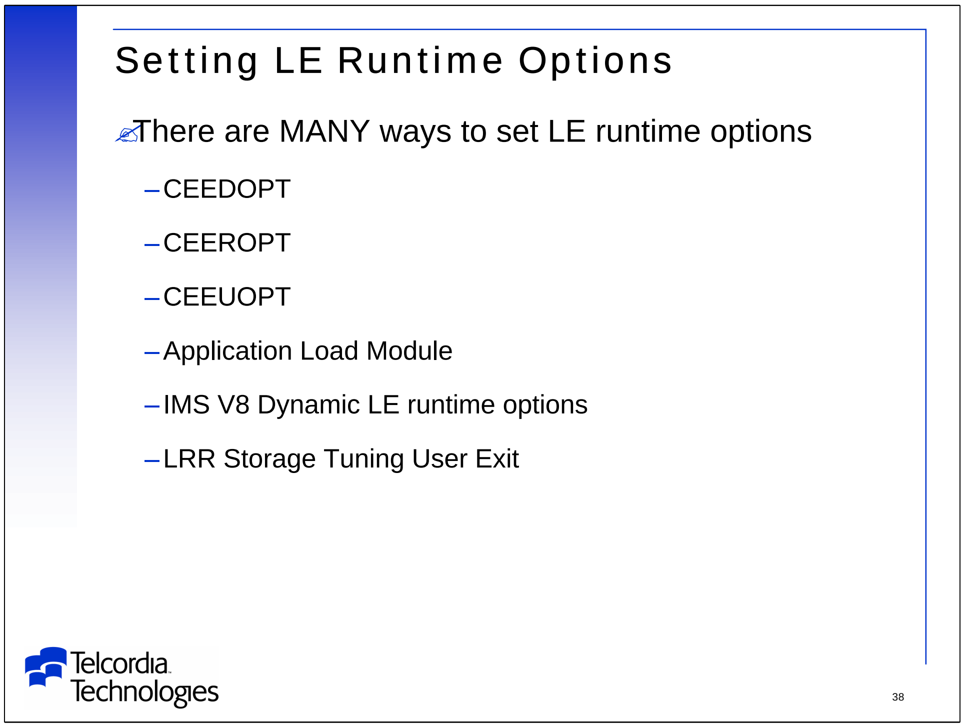
- Favor 31-Bit Application Enhancement
  - IMS applications compiled with C/370 and linked with the pre-LE CTDLI stub and run with ALL31(OFF) may abend because LWS (Library Work Space) storage is not allocated
    - This is fixed with APAR PQ56143
    - Or you can relink with LE version of CTDLI
  - The Reserve Stack needs to be a minimum of 32K
    - STORAGE(,,,nK)
    - Used by LE to process out-of-storage conditions

37

For APAR PQ56143 logic was added back into the C runtime initialization to obtain the LWS area, but ONLY for IMS applications (those linked with CTDLI) and running ALL31(OFF).

-C will obtain the LWS storage only under these circumstances to keep the below-the-line storage at a minimum for most applications.

You can also relink the application with the LE libraries to pick up the new version of CTDLI which does not use the LWS as a module save area.



Setting LE Runtime Options

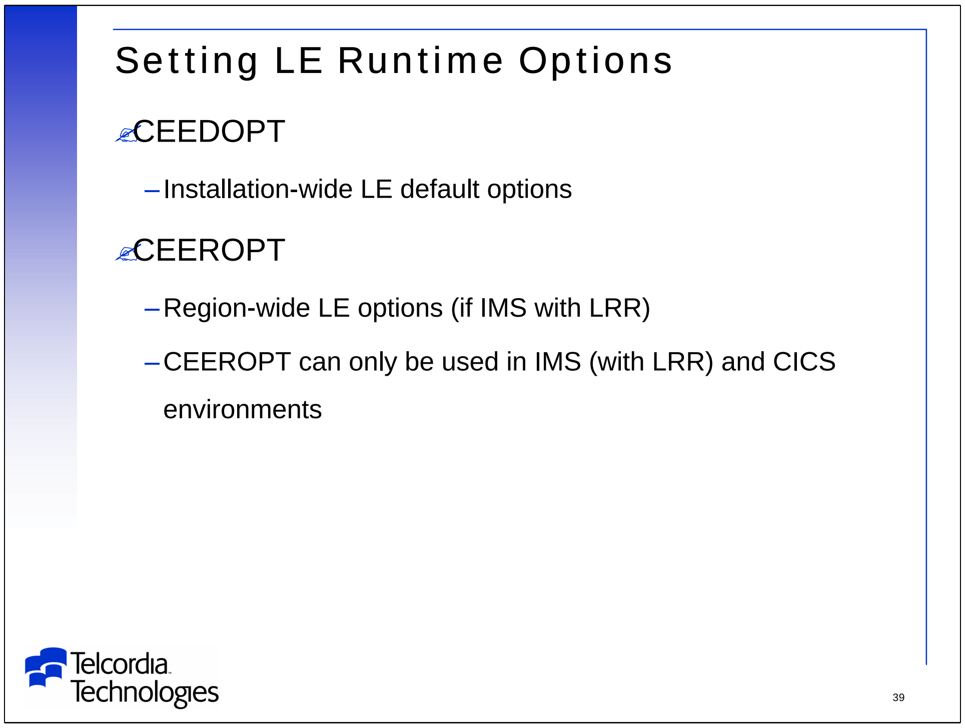
- There are MANY ways to set LE runtime options
  - CEEDOPT
  - CEEROPT
  - CEEUOPT
  - Application Load Module
  - IMS V8 Dynamic LE runtime options
  - LRR Storage Tuning User Exit

Telcordia Technologies

38

There are many ways to set the LE runtime options.

The more specific ways will override the levels above it.



Setting LE Runtime Options

- CEEDOPT
  - Installation-wide LE default options
- CEEROPT
  - Region-wide LE options (if IMS with LRR)
  - CEEROPT can only be used in IMS (with LRR) and CICS environments

Telcordia Technologies

39

CEEDOPT can be used to set installation-wide runtime options, changing the IBM provided defaults.

-The use of CEEDOPT is optional.

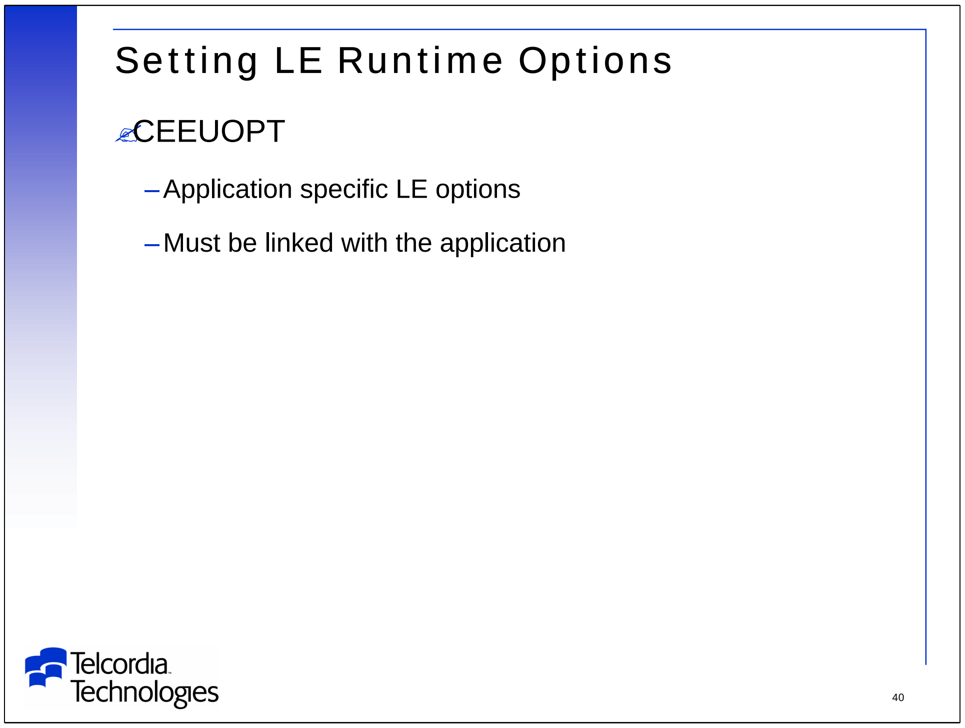
-CEEDOPT does not generate a separate load module, but it does modify several modules in the LE libraries.

CEEROPT can be used to set runtime options for IMS in an IMS dependent region which is also using LRR.

-The use of CEEROPT is optional.

-CEEROPT is a separate load module.

-IMS transactions can be classed so that certain transaction run in certain Message Regions and therefore get certain LE runtime options.



Setting LE Runtime Options

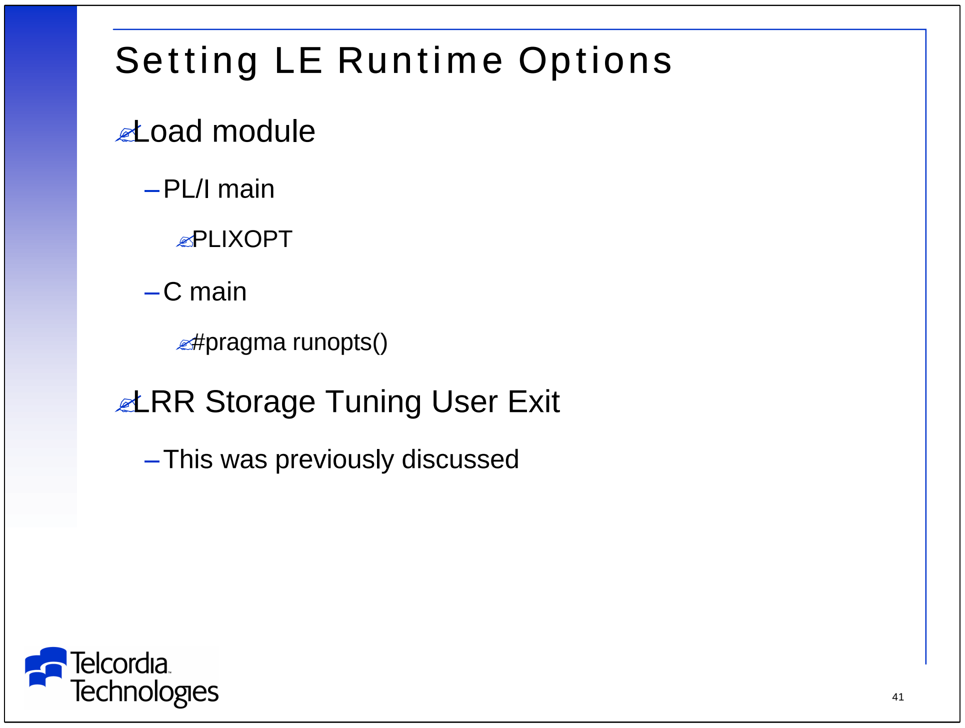
- CEEUOPT
  - Application specific LE options
  - Must be linked with the application

Telcordia Technologies

40

The CEEUOPT CSECT can optionally be included by the application programmer when linking an application to override LE options unless NONOVR was specified in CEEDOPT.

-NONOVR is specified at the option level so that some options may be able to be overridden and some may not.



Setting LE Runtime Options

- Load module
  - PL/I main
    - PLIXOPT
  - C main
    - #pragma runopts()
- LRR Storage Tuning User Exit
  - This was previously discussed

Telcordia Technologies

41

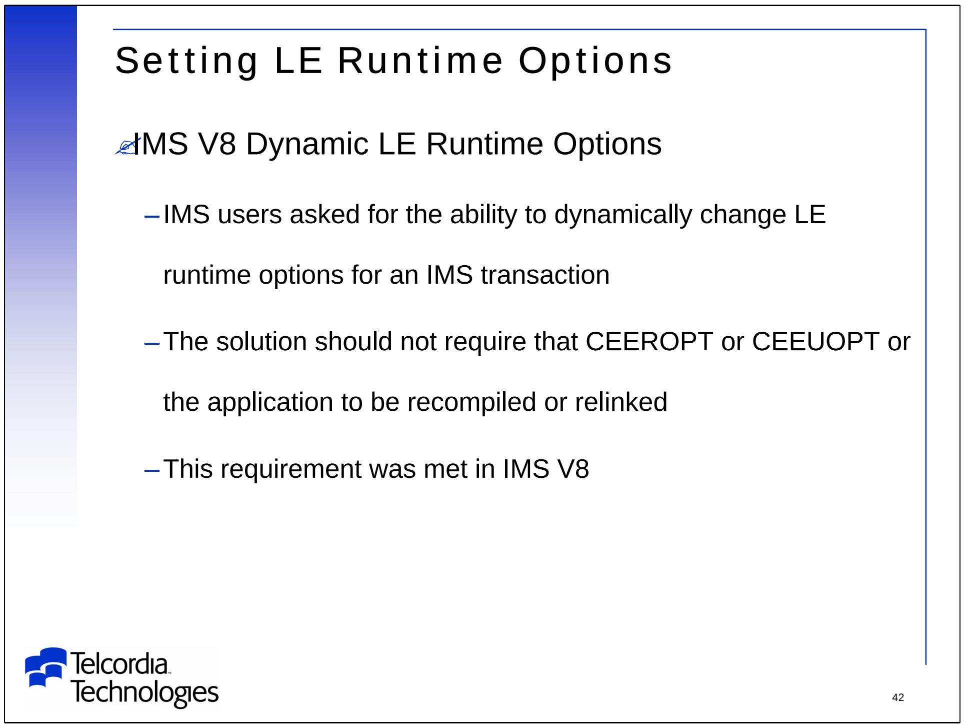
PLIXOPT and #pragma runopts() allow PL/I & C application programs to define runtime options in their source code.

-After compilation these options will be part of the load module.

-PLIXOPT and #pragma runopts() existed prior to LE.

-Pre-LE runtime options will be mapped to equivalent LE runtime options.

-See the LE Migration Guides for details.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - IMS users asked for the ability to dynamically change LE runtime options for an IMS transaction
  - The solution should not require that CEEROPT or CEEUOPT or the application to be recompiled or relinked
  - This requirement was met in IMS V8

Telcordia Technologies

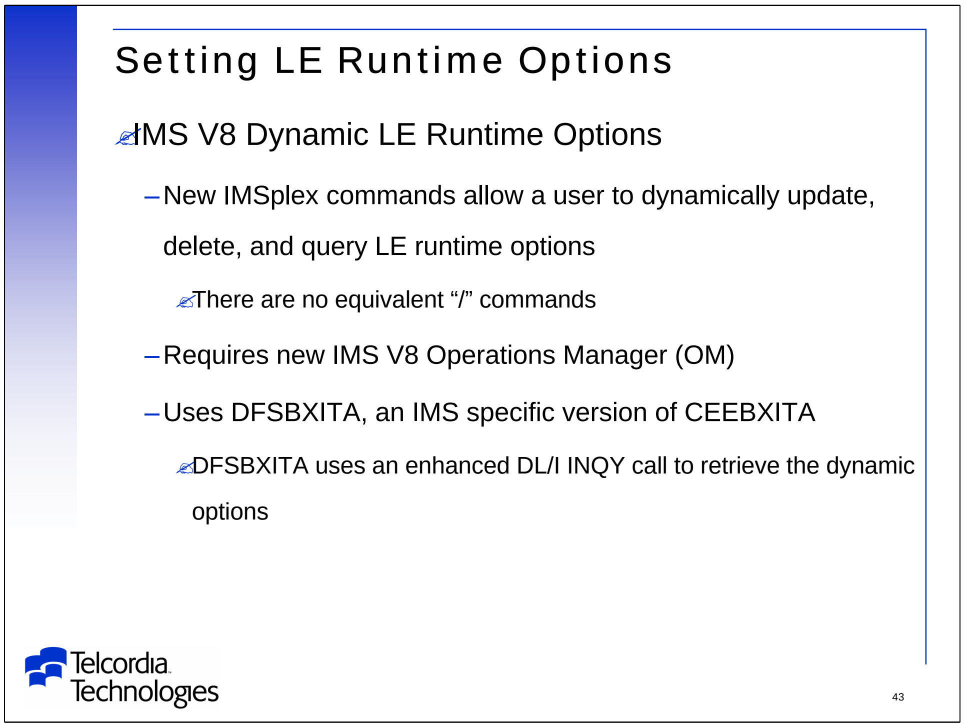
42

IMS users had asked for the ability to dynamically change LE runtime options for an IMS transaction.

This requirement was satisfied in IMS V8.

IMS documentation uses the term “parameters” instead of the LE term “options”.

This presentation will use the term “options”.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - New IMSplex commands allow a user to dynamically update, delete, and query LE runtime options
    - There are no equivalent "/" commands
  - Requires new IMS V8 Operations Manager (OM)
  - Uses DFSBXITA, an IMS specific version of CEEBXITA
    - DFSBXITA uses an enhanced DL/I INQY call to retrieve the dynamic options

Telcordia Technologies

43

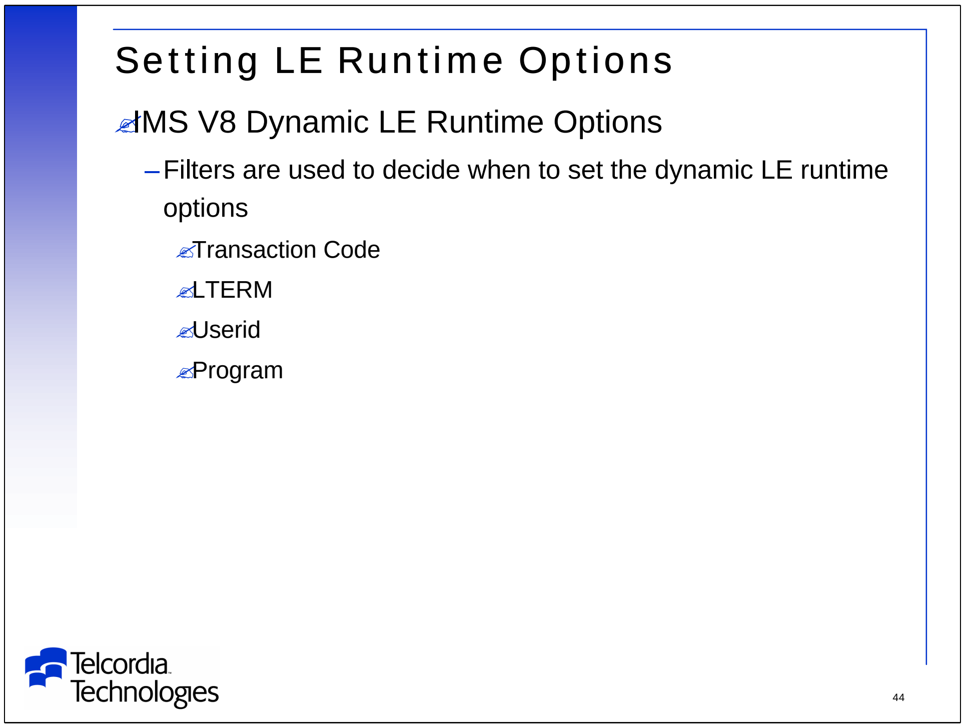
The new IMS V8 Dynamic LE Runtime Options allows LE runtime options to be set when the PSB is scheduled.

DB/DC, DBCTL, and DCCTL environments are supported.

MPP, BMP, IFP, JMP, and JBP regions are supported.

IMS Batch regions are not supported.

-Use CEEROPT for these regions.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - Filters are used to decide when to set the dynamic LE runtime options
    - Transaction Code
    - LTERM
    - Userid
    - Program

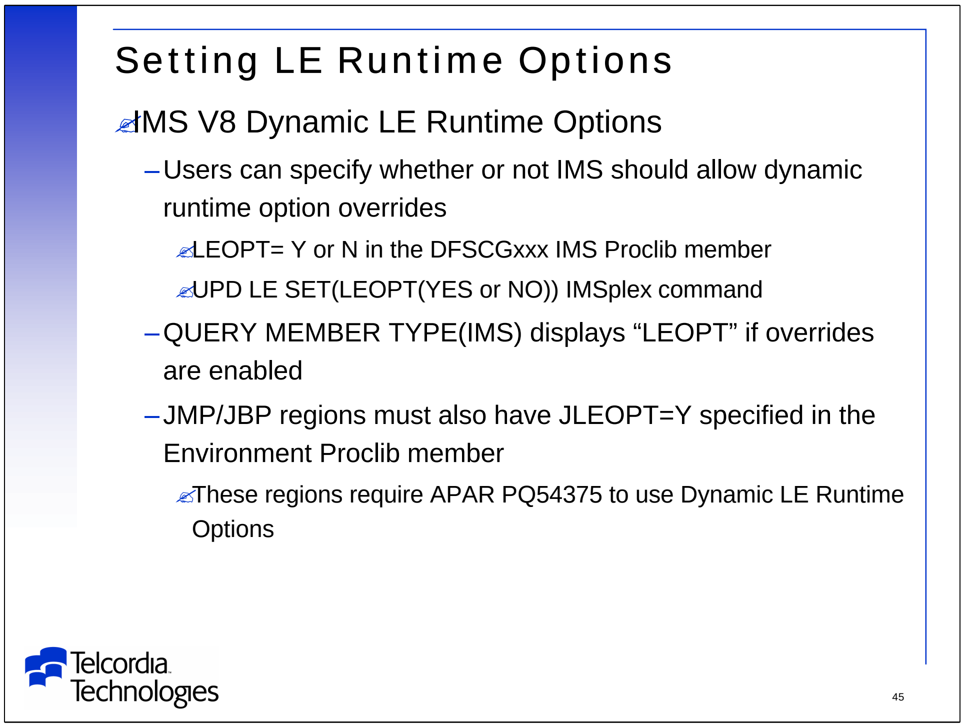
Telcordia Technologies

44

Filters are used to decide when to set the LE options are combinations of Transaction Code, LTERM, Userid, and Program.

-BMP regions only use Program.

As an example, you could set the LE options for a specific IMS transaction only if it was submitted from a specific user from a specific LTERM.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - Users can specify whether or not IMS should allow dynamic runtime option overrides
    - LEOPT= Y or N in the DFSCGxxx IMS Proclib member
    - UPD LE SET(LEOPT(YES or NO)) IMSplex command
  - QUERY MEMBER TYPE(IMS) displays “LEOPT” if overrides are enabled
  - JMP/JBP regions must also have JLEOPT=Y specified in the Environment Proclib member
    - These regions require APAR PQ54375 to use Dynamic LE Runtime Options

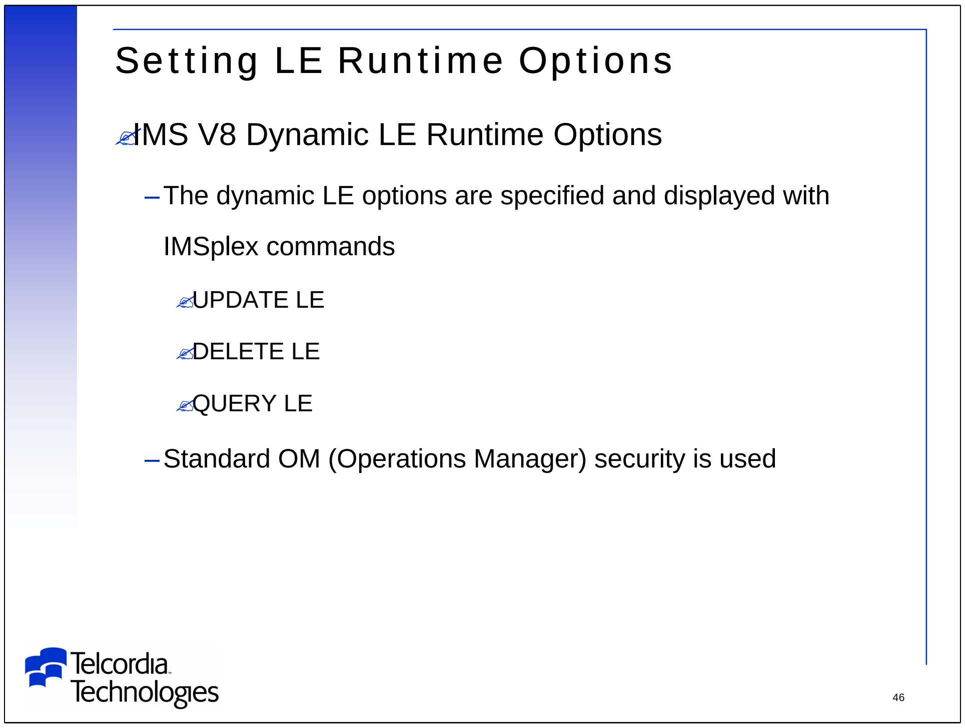
Telcordia Technologies

45

The use of LE dynamic runtime options can be controlled by the LEOPT parameter in DFSCGxxx.

The IMSplex UPD LE SET command can change this option.

JMP/JBP regions require APAR PQ54375 to use Dynamic LE Runtime Options.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - The dynamic LE options are specified and displayed with IMSplex commands
    - UPDATE LE
    - DELETE LE
    - QUERY LE
  - Standard OM (Operations Manager) security is used

Telcordia Technologies

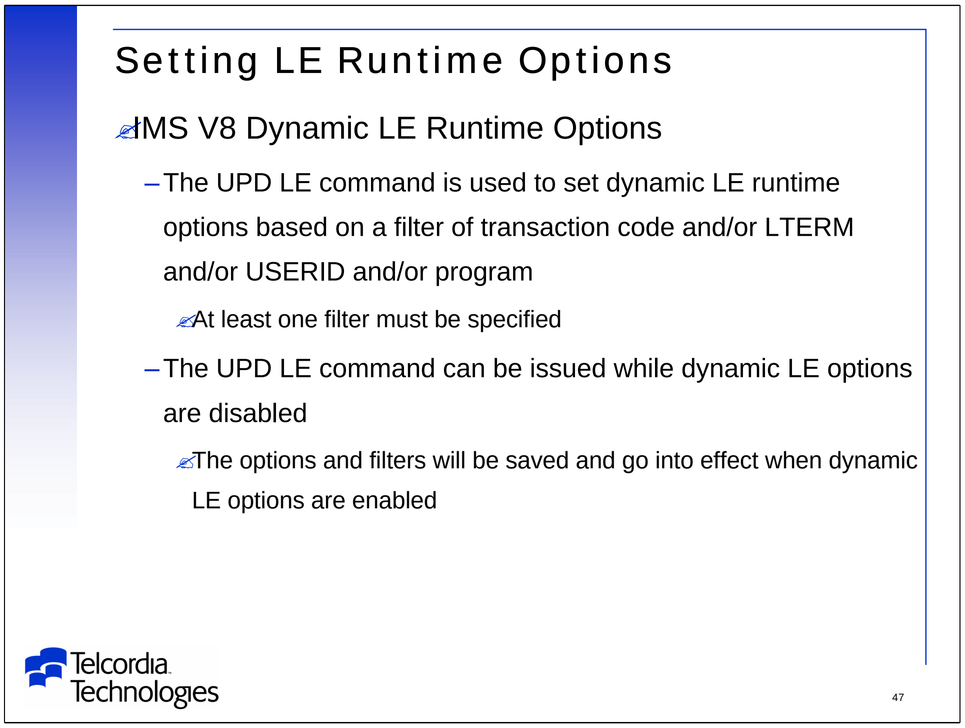
46

The LE runtime options are set , deleted, and displayed using IMSplex commands.

This requires the use of the IMS Operations manager (V8) and an OM client.

This could be the ISPF SPOC (Single Point of Control) provided by IMS.

Standard OM security is used to control access to the commands.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - The UPD LE command is used to set dynamic LE runtime options based on a filter of transaction code and/or LTERM and/or USERID and/or program
    - At least one filter must be specified
  - The UPD LE command can be issued while dynamic LE options are disabled
    - The options and filters will be saved and go into effect when dynamic LE options are enabled

Telcordia Technologies

47

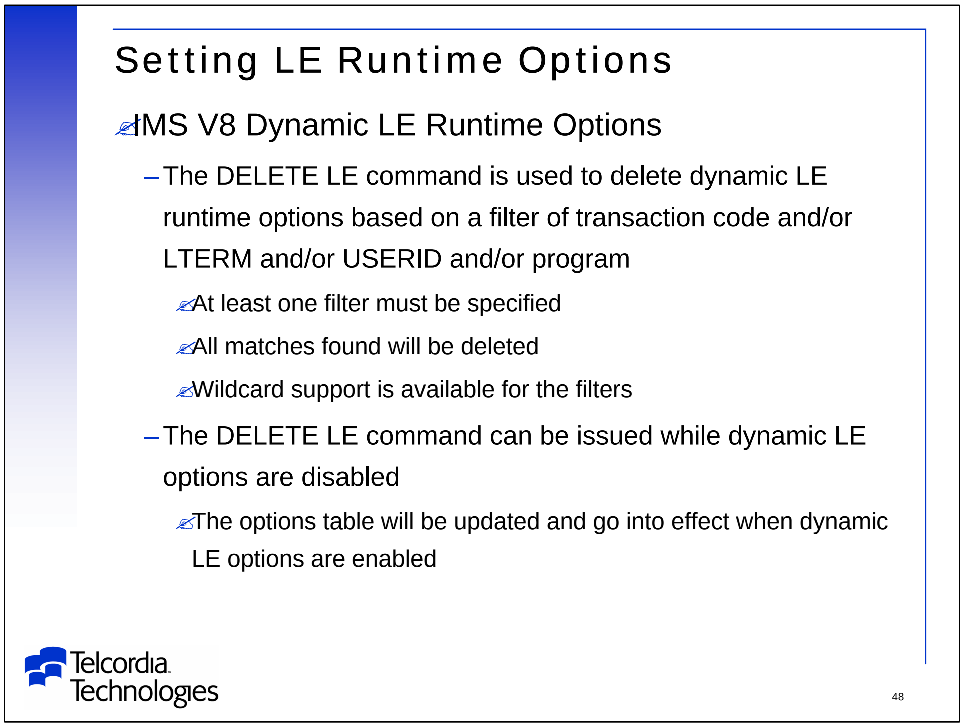
The LE runtime options are set using the UPD LE command, e.g.

```
UPD LE TRAN(PART) LTERM(TERM2) SET(LERUNOPTS(xxxx))
UPD LE PGM(DFSSAM02) SET(LERUNOPTS(yyyy))
```

The dynamic runtime options are stored in a table.

The UPD LE command can be issued even when the use of dynamic LE runtime options has been disabled.

The table will still be updated and the options will go into effect when dynamic LE runtime options is enabled.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - The DELETE LE command is used to delete dynamic LE runtime options based on a filter of transaction code and/or LTERM and/or USERID and/or program
    - At least one filter must be specified
    - All matches found will be deleted
    - Wildcard support is available for the filters
  - The DELETE LE command can be issued while dynamic LE options are disabled
    - The options table will be updated and go into effect when dynamic LE options are enabled

Telcordia Technologies

48

The LE runtime options are deleted using the DELETE LE command, e.g.

DELETE LE TRAN(PART) LTERM(TERM2)

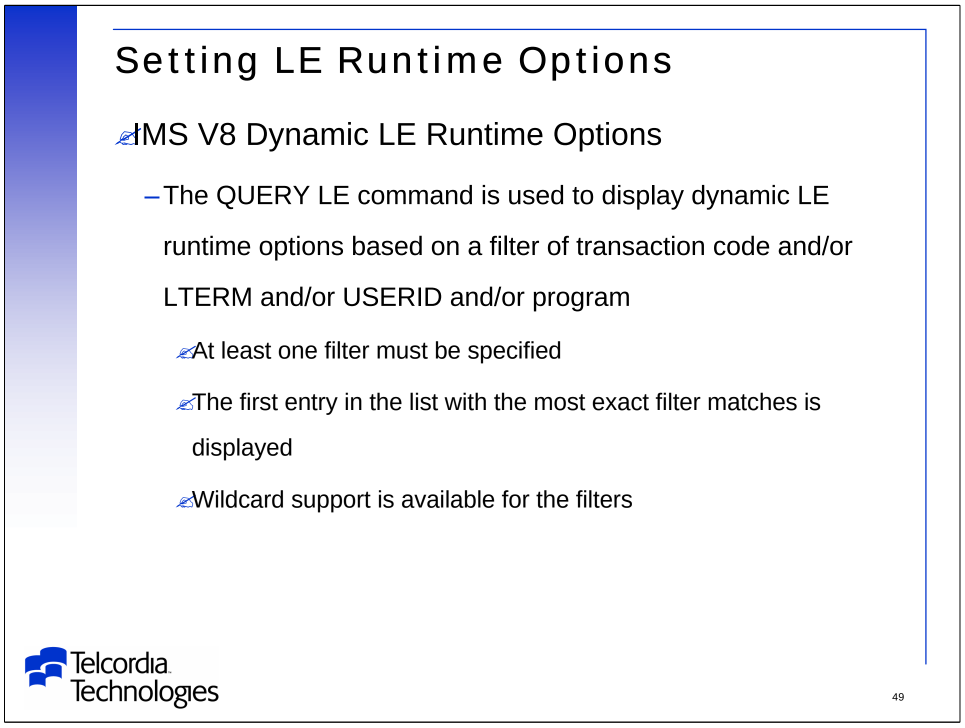
DELETE LE PGM(DFSSAM02)

DELETE LE USERID(STEVE*)

The DELETE LE command can be issued even when the use of dynamic LE runtime options has been disabled.

The table will still be updated and go into effect when dynamic LE runtime options is enabled.

There is wildcard support for the TRAN, LTERM, USERID, and PGM values.



Setting LE Runtime Options

- IMS V8 Dynamic LE Runtime Options
  - The QUERY LE command is used to display dynamic LE runtime options based on a filter of transaction code and/or LTERM and/or USERID and/or program
    - At least one filter must be specified
    - The first entry in the list with the most exact filter matches is displayed
    - Wildcard support is available for the filters

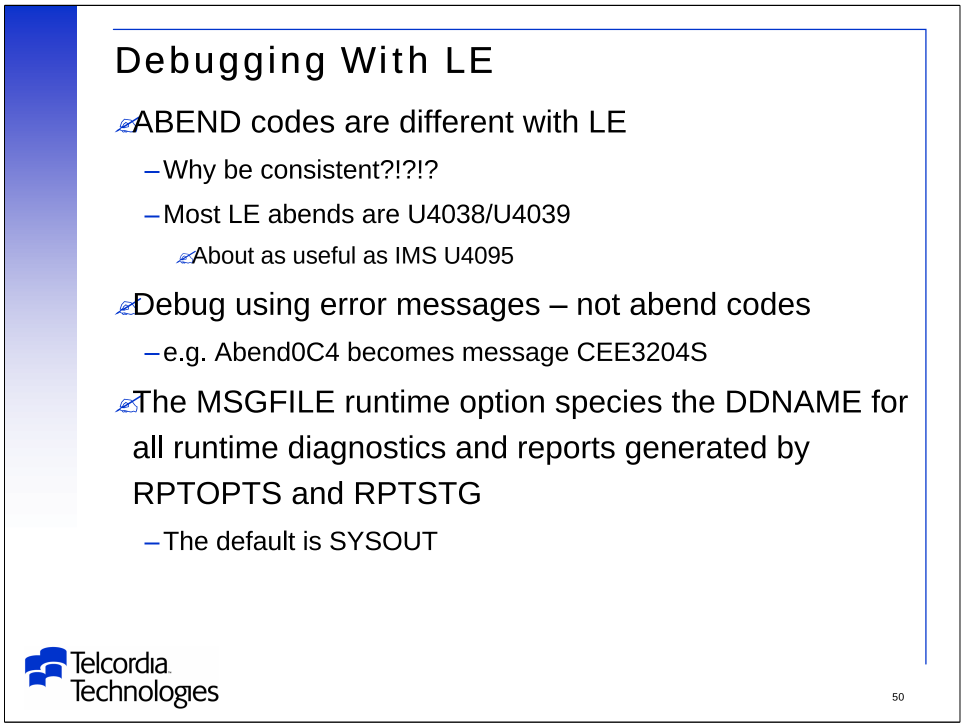
Telcordia Technologies

49

The LE runtime options are displayed using the QUERY LE command, e.g.

```
QUERY LE TRAN(PART) SHOW(ALL)
 QUERY LE USERID(STEVE*) SHOW(ALL)
```

There is wildcard support for the TRAN, LTERM, USERID, and PGM values.



Debugging With LE

- ABEND codes are different with LE
  - Why be consistent?!?!?
  - Most LE abends are U4038/U4039
    - About as useful as IMS U4095
- Debug using error messages – not abend codes
  - e.g. Abend0C4 becomes message CEE3204S
- The MSGFILE runtime option species the DDNAME for all runtime diagnostics and reports generated by RPTOPTS and RPTSTG
  - The default is SYSOUT

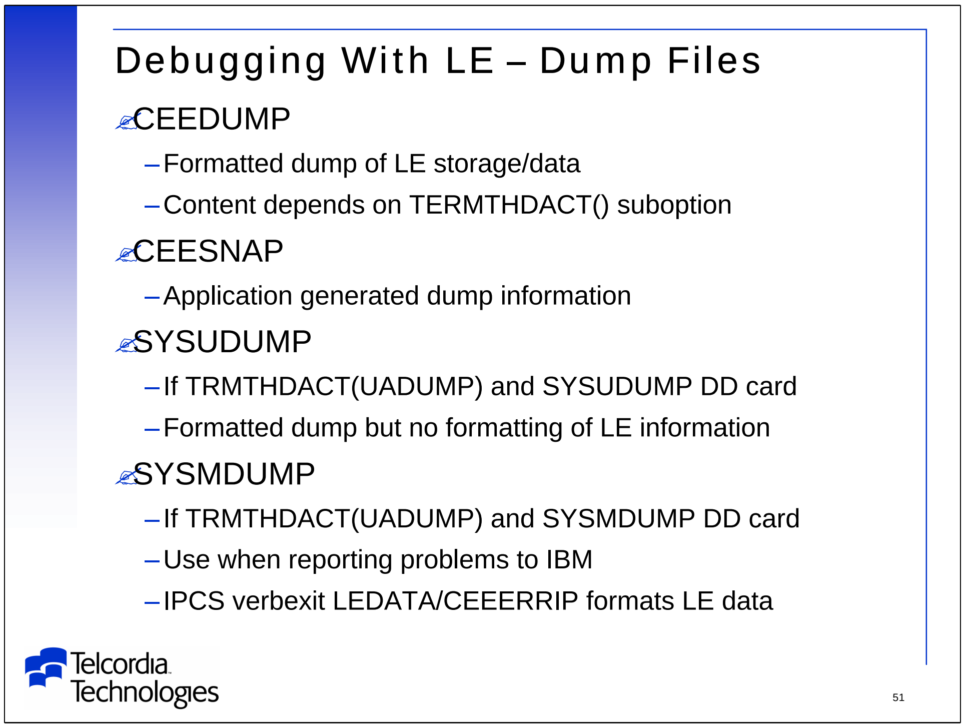
Telcordia Technologies

50

Just when you got used to U4000 abends – they changed the number.

U4038 will be used if there is no dump.

U4039 will be used if there is a dump.



Debugging With LE – Dump Files

- CEEDUMP
  - Formatted dump of LE storage/data
  - Content depends on TERMTHDACT() suboption
- CEESNAP
  - Application generated dump information
- SYSUDUMP
  - If TRMTHDACT(UADUMP) and SYSUDUMP DD card
  - Formatted dump but no formatting of LE information
- SYSMDUMP
  - If TRMTHDACT(UADUMP) and SYSMDUMP DD card
  - Use when reporting problems to IBM
  - IPCS verbexit LEDATA/CEEERRIP formats LE data

Telcordia Technologies

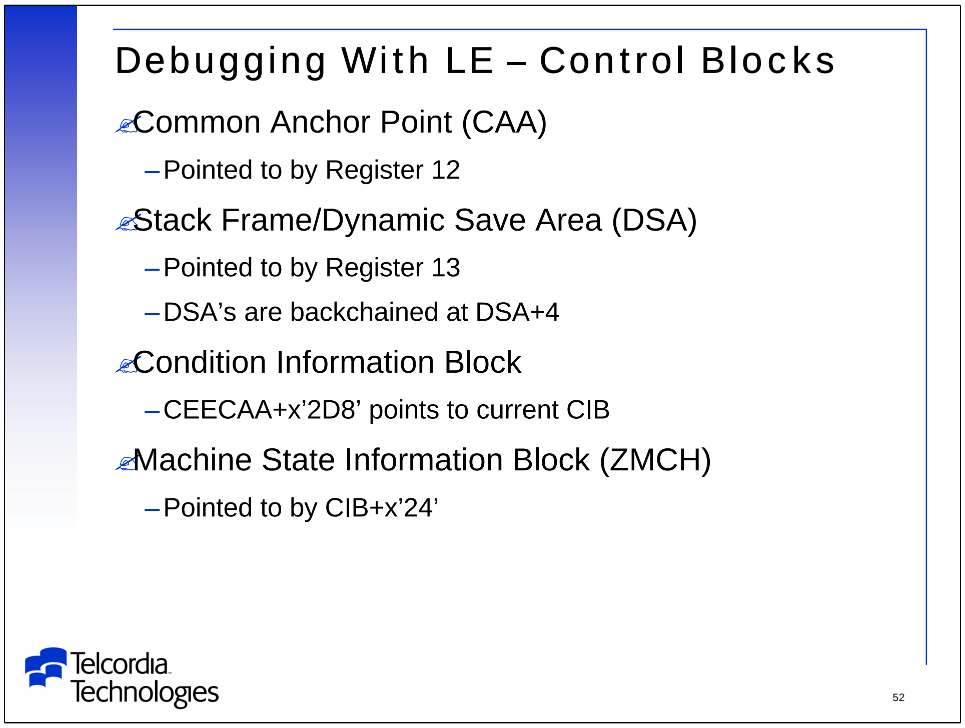
51

Ensure you have the proper DD cards for capturing debugging and problem resolution data.

If your routine is running under z/OS or CICS, you can generate useful diagnostic information by using the cdump() function or PLIDUMP.

-This produces a main storage dump with the activation stack.

-This is equivalent to calling CEE3DMP with the option string: TRACEBACK BLOCKS VARIABLES FILES STORAGE STACKFRAME(ALL) CONDITION ENTRY.



Debugging With LE – Control Blocks

- Common Anchor Point (CAA)
  - Pointed to by Register 12
- Stack Frame/Dynamic Save Area (DSA)
  - Pointed to by Register 13
  - DSA's are backchained at DSA+4
- Condition Information Block
  - CEECAA+x'2D8' points to current CIB
- Machine State Information Block (ZMCH)
  - Pointed to by CIB+x'24'

Telcordia Technologies

52

Each LE ***thread*** is represented by a common anchor area (CAA) which is the 'SCD' of LE.

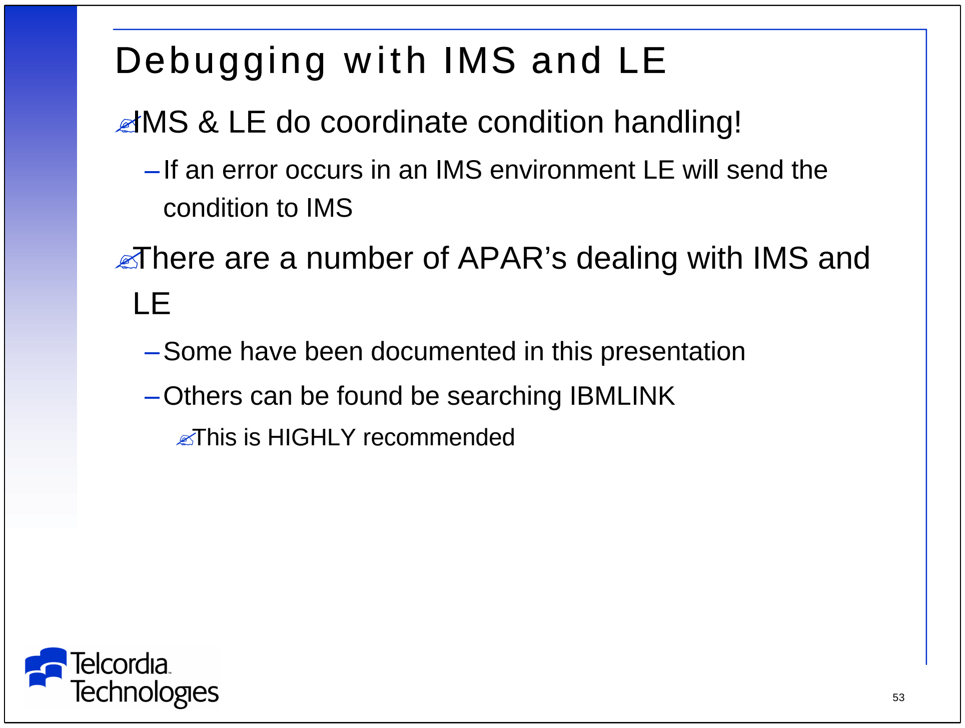
A DSA is acquired every time a separately compiled procedure or block is entered.

-A stack frame is also allocated for each call to a LE service.

-The first DSA on the chain is a "Dummy DSA".

The LE condition manager creates a CIB for each condition encountered in the LE environment.

The LE machine state information block (ZMCH) contains information about the hardware state (PSW & registers) at the time of the error for program checks.



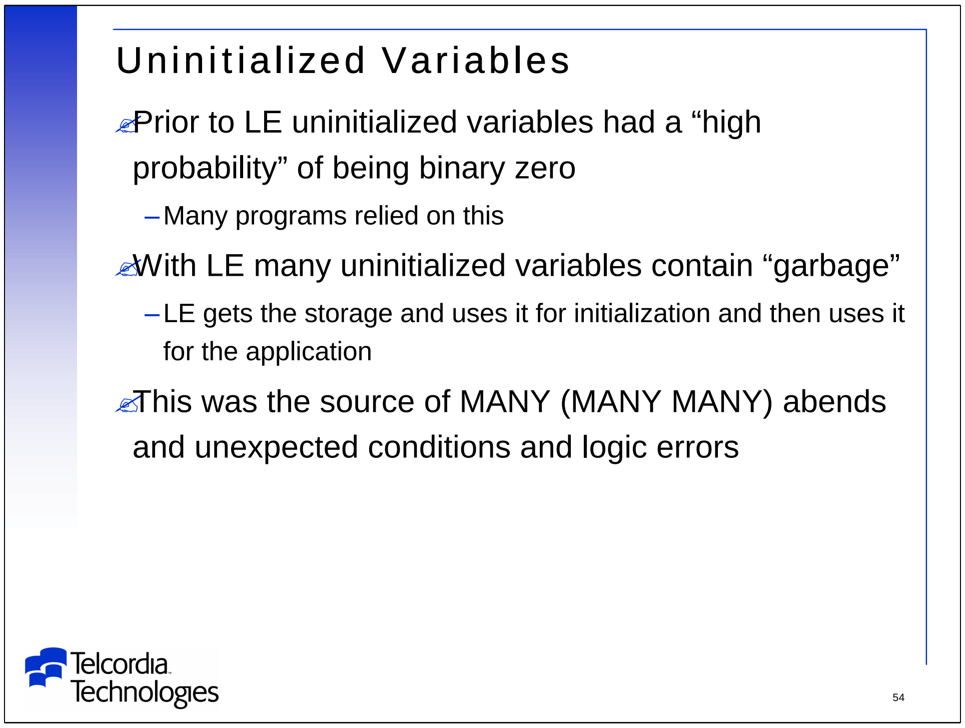
Debugging with IMS and LE

- IMS & LE do coordinate condition handling!
  - If an error occurs in an IMS environment LE will send the condition to IMS
- There are a number of APAR's dealing with IMS and LE
  - Some have been documented in this presentation
  - Others can be found be searching IBMLINK
    - This is HIGHLY recommended

Telcordia Technologies

53

If a program interrupt or abend occurs when your application is running in an IMS environment LE percolates the condition back to IMS.



Uninitialized Variables

- Prior to LE uninitialized variables had a “high probability” of being binary zero
  - Many programs relied on this
- With LE many uninitialized variables contain “garbage”
  - LE gets the storage and uses it for initialization and then uses it for the application
- This was the source of MANY (MANY MANY) abends and unexpected conditions and logic errors

Telcordia Technologies

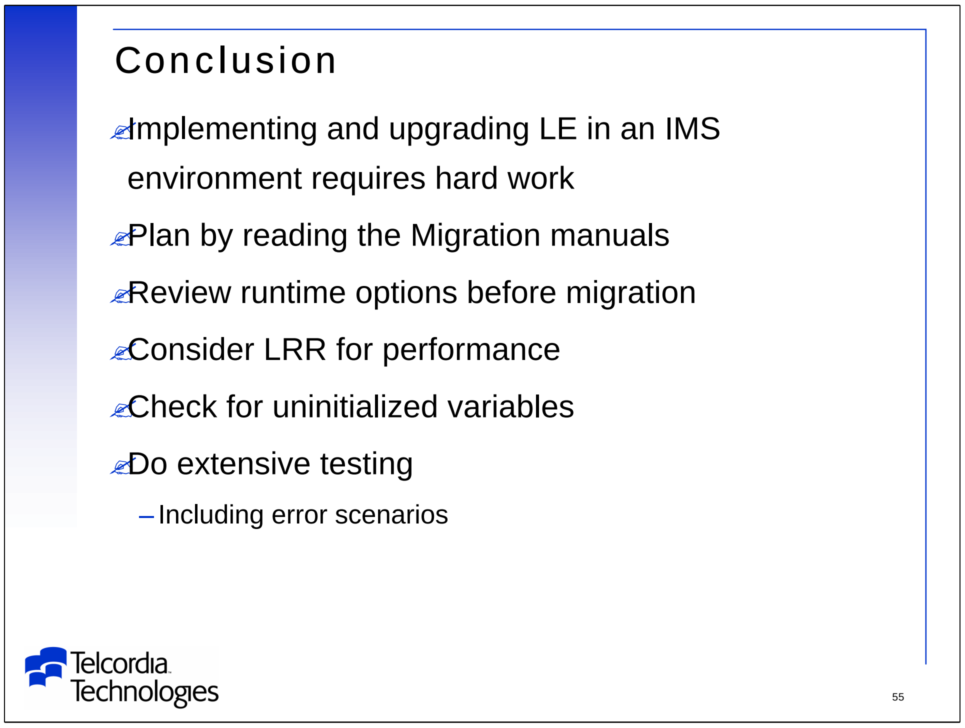
54

In an LE environment uninitialized variables can easily contain “garbage” – other than x’00’.

We have many applications which assumed (or were lucky) that this data started as x’00’ in a pre-LE environment.

It took a great deal of time and energy to find and fix all of the problems.

-There may still be some surprises waiting to be found.



Conclusion

- Implementing and upgrading LE in an IMS environment requires hard work
- Plan by reading the Migration manuals
- Review runtime options before migration
- Consider LRR for performance
- Check for uninitialized variables
- Do extensive testing
  - Including error scenarios

Telcordia Technologies

55

I am sure there are other considerations for LE and IMS.

We have not tested all languages and options.

Check IBMLINK regularly.

Keep up to date with IMS maintenance.

The IMS LISTSERV and Assembler LISTSERV are great place for more information.

Questions?

56