z/OS

Language Environment Run-Time Application Migration Guide

GA22-7565-12

Note
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 33.

This edition applies to Language Environment® in version 1, release 13, modification 0 of z/OS (product number 5694-A01), and to subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces GA22-7565-11.

Contents

Tables

About this document

This document supports z/OS (5694-A01).

IBM® z/OS Language Environment (also called Language Environment) provides common services and language-specific routines in a single run-time environment for C, C++, COBOL, Fortran (z/OS only; no support for z/OS UNIX System Services or CICS®), PL/I, and assembler applications. It offers consistent and predictable results for language applications, independent of the language in which they are written.

Language Environment is the prerequisite run-time environment for applications generated with the following IBM compiler products:
- z/OS XL C/C++ (feature of z/OS)
- z/OS C/C++
- OS/390 C/C++
- C/C++ for MVS/ESA
- C/C++ for z/VM
- XL C/C++ for z/VM
- AD/Cycle® C/370™
- VisualAge for Java, Enterprise Edition for OS/390
- Enterprise COBOL for z/OS
- Enterprise COBOL for z/OS and OS/390
- COBOL for OS/390 & VM
- COBOL for MVS & VM (formerly COBOL/370)
- Enterprise PL/I for z/OS
- Enterprise PL/I for z/OS and OS/390
- VisualAge PL/I
- PL/I for MVS & VM (formerly PL/I MVS™ & VM)
- VS FORTRAN and FORTRAN IV (in compatibility mode)

Although not all compilers listed are currently supported, Language Environment supports the compiled objects that they created.

Language Environment supports, but is not required for, an interactive debug tool for debugging applications in your native z/OS environment.

Debug Tool is also available as a standalone product. Debug Tool Utilities and Advanced Functions is also available. For more information, see http://www.ibm.com/software/awdtools/debugtool/.

Language Environment supports, but is not required for, VS FORTRAN Version 2 compiled code (z/OS only).

Language Environment consists of the common execution library (CEL) and the run-time libraries for C/C++, COBOL, Fortran, and PL/I.

For more information on VisualAge for Java, Enterprise Edition for OS/390, program number 5655-JAV, see the product documentation.

This book provides an overview of the steps z/OS customers must take to migrate applications for use with z/OS Language Environment. These customers may not necessarily be migrating to a new language compiler.

This book is written for application developers. Familiarity with the run-time libraries of the different languages, and an understanding of the basics of linking and running applications, are assumed.

The information in this book will not provide a comprehensive guide to the migration process; rather, it is designed to help you create a broad migration strategy. This book will help you identify which modules can be migrated first, and which will require relinking or recompiling. It also explains how to use Language Environment run-time options to achieve behavior that is compatible with your old modules. For more detailed information about migration topics such as upgrading source code and load module compatibility, see one of the following manuals:

- *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*
- *IBM C for VM/ESA Compiler and Run-Time Migration Guide*
- *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409.
- *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide*
- *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide*
- *Fortran Run-Time Migration Guide*

Using your documentation

The publications provided with Language Environment are designed to help you:

- Manage the run-time environment for applications generated with a Language Environment-conforming compiler.
- Write applications that use the Language Environment callable services.
- Develop interlanguage communication applications.
- Customize Language Environment.
- Debug problems in applications that run with Language Environment.
- Migrate your high-level language applications to Language Environment.

Language programming information is provided in the supported high-level language programming manuals, which provide language definition, library function syntax and semantics, and programming guidance information.

Each publication helps you perform different tasks, some of which are listed in Table 1. All books are available in printable (PDF) and BookManager softcopy formats. For a complete list of publications that you may need, see "Bibliography" on page 37.

Table 1. How to Use z/OS Language Environment Publications

To ...	Use ...
Evaluate Language Environment	*z/OS Language Environment Concepts Guide*
Plan for Language Environment	*z/OS Language Environment Concepts Guide* *z/OS Language Environment Run-Time Application Migration Guide*
Install Language Environment	*z/OS Program Directory*
Customize Language Environment	*z/OS Language Environment Customization*

Table 1. How to Use z/OS Language Environment Publications (continued)

To ...	Use ...
Understand Language Environment program models and concepts	*z/OS Language Environment Concepts Guide* *z/OS Language Environment Programming Guide* *z/OS Language Environment Programming Guide for 64-bit Virtual Addressing Mode*
Find syntax for Language Environment run-time options and callable services	*z/OS Language Environment Programming Reference*
Develop applications that run with Language Environment	*z/OS Language Environment Programming Guide* and your language programming guide
Debug applications that run with Language Environment, diagnose problems with Language Environment	*z/OS Language Environment Debugging Guide*
Get details on run-time messages	*z/OS Language Environment Run-Time Messages*
Develop interlanguage communication (ILC) applications	*z/OS Language Environment Writing Interlanguage Communication Applications* and your language programming guide
Migrate applications to Language Environment	*z/OS Language Environment Run-Time Application Migration Guide* and the migration guide for each Language Environment-enabled language

Where to find more information

For an overview of the information associated with z/OS, see *z/OS Information Roadmap*.

Information updates on the web

For the latest information updates that have been provided in PTF cover letters and documentation APARs for z/OS®, see the online document at: http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ZDOCAPAR

This document is updated weekly and lists documentation changes before they are incorporated into z/OS publications.

The z/OS Basic Skills Information Center

The z/OS Basic Skills Information Center is a Web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. The Information Center is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS system programmer.

Specifically, the z/OS Basic Skills Information Center is intended to achieve the following objectives:

- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS.

To access the z/OS Basic Skills Information Center, open your Web browser to the following Web site, which is available to all users (no login required):
http://publib.boulder.ibm.com/infocenter/zos/basics/index.jsp

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

1. Send an email to mhvrcfs@us.ibm.com
2. Visit the Contact z/OS web page at http://www.ibm.com/systems/z/os/zos/webqs.html
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Building 707
   2455 South Road
   Poughkeepsie, NY 12601-5400
   U.S.A.
4. Fax the comments to us as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:

- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number:
  z/OS V1R13.0 Language Environment Run-Time Application Migration Guide
  GA22-7565-12
- The topic and page number related to your comment
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:

- Contact your IBM service representative
- Call IBM technical support
- Visit the IBM support portal at http://www.ibm.com/systems/z/support/

Summary of changes

This document contains terminology, maintenance, and editorial changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Changes made in z/OS Version 1 Release 13

This document contains information that was previously presented in *z/OS Language Environment Run-Time Application Migration Guide*, GA22-7565-11, which supports z/OS Version 1 Release 12.

Deleted information:

- The table on Language Environment Run-Time Options, Defaults, and Recommendations has been moved to *z/OS Language Environment Customization*
- Appendix A, Language Environment-enabled vendor tools and application packages, has been deleted.

Changes made in z/OS Version 1 Release 12

This document contains information that was previously presented in *z/OS Language Environment Run-Time Application Migration Guide*, GA22-7565-10, which supports z/OS Version 1 Release 11.

New information:

- "Setting run-time options as overrideable or nonoverrideable" on page 6 describes the change in setting run-time options as overrideable or nonoverrideable in the CEEPRMxx parmlib member.
- "Changes to the Language Environment run-time options report when issuing D CEE" on page 7 describes the changes made to the run-time options report when a D CEE command is issued.
- "Removal of conversion table source code" on page 8 describes removal of the source code for conversion tables.
- "Changes to the CICS CLER run-time options report" on page 9 describes the change to the run-time options report displayed from the CICS CLER transaction.

Changed information:

- The "Readers' Comments - We'd Like to Hear from You" section at the back of this publication has been replaced with a new section "How to send your comments to IBM" on page xi. The hardcopy mail-in form has been replaced with a page that provides information appropriate for submitting readers comments to IBM.

Changes made in z/OS Version 1 Release 11

This document contains information that was previously presented in *z/OS Language Environment Run-Time Application Migration Guide*, GA22-7565-09, which supports z/OS Version 1 Release 10.

Changed information:

- HEAPCHK(ON,0,0,10) changed to the new default values (HEAPCHK(OFF,1,0,0,0,1024,0,1024,0)).
- Binary and Decimal Floating-Point support in the CICS Environment.

Chapter 1. Planning to migrate to Language Environment

This topic provides a checklist to help you plan the migration of your applications to the Language Environment run-time environment for the first time.

For more detailed information about migration considerations, see Chapter 3, "Migrating from other run-time environments," on page 15. If you are migrating from a previous release of Language Environment, you should review the information in Chapter 2, "Migrating from another Language Environment release," on page 5.

Checklist for migration

Each task in the following checklist is recommended; you should perform each task in the order shown.

1. **Learn about Language Environment.**

   Ensure that you and other application programmers who will be involved in the migration effort are familiar with the features of Language Environment and the differences between your current run-time environment and the Language Environment run-time environment. You can get information about Language Environment from publications such as:
   - *z/OS Language Environment Customization*
   - *z/OS Language Environment Concepts Guide*

2. **Take an inventory of the applications and vendor products you intend to run with Language Environment.**
   - C, C++, COBOL, Fortran, PL/I, or Assembler programs

     For each program you intend to move to the Language Environment run-time environment, obtain the following information:
     - Version and release of the compiler that generated the program
     - Which COBOL programs were compiled with RES and which with NORES
     - Run-time options used and how they were specified
     - Which PL/I programs use the shared library and which ones do not
     - Which programs call, or are called by, assembler programs
     - Which applications contain interlanguage communication (ILC)
     - Which programs are used with CICS, IMS™, DB2®, or other subsystems
     - Control statements used
     - Frequency and types of abends
     - Test cases required and available
     - Amount of storage used
     - Frequency of execution of reusable or common modules
     - Program execution time (processor (CPU) and elapsed)
   - Vendor tools, packages, and products
     - Ensure that all vendor tools, packages, and products run with Language Environment; any source code for the packages must also be compatible with your Language Environment-conforming compiler.
     - Ensure that any vendor code generators generate code that is compatible with your Language Environment-conforming compiler.
     - Ensure that vendor development tools and debuggers will not issue their own ESPIE or ESTAE, as Language Environment must get control first.

3. **Prioritize programs.**

   Determine the effort required to migrate each program and the order in which you will migrate them. Each program will require some level of effort to migrate, ranging from minimal testing to a code rewrite. Using the information from your inventory analysis, determine if each program:
   - Requires minimum, moderate, or extensive testing
   - Runs with Language Environment without change
   - Requires relinking with Language Environment
   - Must be recompiled with a Language Environment-conforming compiler, without change to the source code
   - Requires changes to the source code
   - Does not run with Language Environment

   After you have determined the effort required to migrate each load module, list your programs in the order you want to move them to Language Environment. You should consider the importance of each program and how often it is used.

   You should migrate applications that contain ILC after you have migrated any applications that contain only C, C++, COBOL, Fortran, or PL/I. (An application that contains assembler, but is otherwise created from one language, is not considered an ILC application in this information.) For information about compatibility considerations for ILC applications, see "Migrating ILC applications to Language Environment" on page 16.

4. **Install Language Environment.**

   Perform the following tasks, which can be done concurrently:
   - Change default run-time options as appropriate.

     To ensure that the Language Environment run-time results are compatible with your current run-time results, you will need to change some of the default settings for the run-time options. For a list of recommended settings, see *z/OS Language Environment Customization*.
   - Assess storage requirements.

     Storage requirements may be larger for Language Environment than for your current run-time environment. During conversion, you might need DASD for the Language Environment run-time library and for the run-time library that you are currently using. For information about Language Environment DASD requirements, see *z/OS Program Directory*.

     Virtual storage requirements for placing library routines above or below the 16M line may also increase, depending on which Language Environment storage options you specify. For recommended settins, see *z/OS Language Environment Customization*.
   - Determine how to phase-in the Language Environment run-time environment using a STEPLIB approach or by adding Language Environment to the LNKLST.

     Using the STEPLIB approach, you can gradually phase-in the Language Environment run-time environment. When you use STEPLIB statements to specify the Language Environment run-time environment, you can phase-in one region (CICS or IMS), batch (group of applications), or user (TSO) at a time. Although using STEPLIB means changing JCL, a gradual conversion can be easier than moving all of your applications at one time.

     When you add Language Environment to the LNKLST, it is available to all of your applications. Ensure all applications are functioning correctly with Language Environment before adding Language Environment to your LNKLST. You might consider temporarily adding Language Environment to the LNKLST until you have confirmed the applications work as intended.

5. **Set up a regression testing procedure.**

   To ensure that the Language Environment run-time results are compatible with your current run-time results, you will need to perform regression tests on all the programs you migrate. Run your applications in parallel with your current run-time environment and with the Language Environment run-time environment to confirm that the results are the same. You can temporarily add Language Environment to the LINKLST to accomplish this. When your applications are running with Language Environment in a test environment, you should take performance measurements, especially on any time-critical or response-critical applications.

6. **Move applications into production.**

   When your testing shows the entire application (or group of applications, if running more than one application in an IMS region or under TSO) runs as expected, you can move the entire unit over to production use. However, if an unexpected error occurs, you may need to perform one of the following steps:

   - On z/OS systems, run the previous version of your application as a substitute.
   - Under DB2, CICS, and IMS, return to the last commit point and then continue processing from that point using the previous version of the program. For DB2, use an SQL ROLLBACK WORK statement.
   - For batch applications, use the backup and restore facilities at your site to recover.

   After you move your applications to production use with the Language Environment run-time environment, monitor your applications to ensure that they continue to work properly. You can then run with the confidence that you had in your previous run-time library.

Planning to link and run with Language Environment

Language Environment provides separate libraries for linking and running applications. The link libraries, of which SCEELKED is one, contain static (resident) routines that are linked with the application and used to resolve external references at link-edit time. The load library, of which SCEERUN is one, contains dynamic routines that are not part of the application and are dynamically loaded at run time. Language Environment callable services and other routines, such as those for initialization and termination, are located in SCEERUN. For a complete list of libraries and which phase of application development they are used in, see *z/OS Language Environment Customization*.

You will need to modify the job control statements in your input stream to point directly to SCEELKED and SCEERUN, or to point to the appropriate IBM-supplied cataloged procedures, if your job uses cataloged procedures. See *z/OS Language Environment Programming Guide* for more information about linking and running and using cataloged procedures.

On z/OS systems, you can install reentrant members of the SCEERUN data set in the link pack area (LPA) for faster retrieval. IBM provides a data set called SCEELPA and highly recommends putting this data set in the LPA (or LPALSTxx). This data set contains modules that are reentrant, reside above the line, and are heavily used by z/OS itself.

Chapter 2. Migrating from another Language Environment release

This topic provides information about migrating from one release of Language Environment to another. This topic explains upward compatibility as well as downward compatibility. Please look at any migration considerations listed in this topic for the release you are migrating to as well as any releases you are skipping over.

Language Environment provides general object and load module compatibility for applications that ran with a previous release of Language Environment. All Language Environment-enabled applications that have been linked with a minimum level of Language Environment for MVS and VM 1.3 will continue to run with later releases of Language Environment without the need to relink the application. If you experience any problems (for example, an application that worked with Language Environment for MVS and VM 1.3 no longer works after you install the current release of Language Environment), you should report them to IBM.

Most load modules are compatible with any level of Language Environment that is equivalent to, or higher than, the level used to link-edit them. Similarly, object modules can be link-edited with any level of Language Environment that is equivalent to, or higher than, the level required by the compiler that generated them.

Please see any exceptions in this topic.

Migration actions required for each release

This topic describes common activities and considerations that are typically required when you migrate from one release of Language Environment to another:

1. Update the CICS System Definition (CSD) file using the program definitions in the CEECCSD member (and CEECCSDX member for CICS TS 3.1) found in the SCEESAMP data set. Language Environment may have changed (added or deleted) load modules in this release.
2. Update the Language Environment load modules that were placed in the link pack area (LPA). Sample members found in the SCEESAMP data set, which can be used to move load modules into LPA, should be reviewed with every release migration. See the table "Language Environment sample IEALPAnn or PROGxx Members in CEE.SCEESAMP" for the list of sample members and their changed content in *z/OS Language Environment Customization*.
3. If any Language Environment user exits were used at the previous release and you plan to use them with the new release, they must be re-linked using the new release of Language Environment.
4. Updates to sample jobs related to run-time options might be required.

   Changes are required to the CEEWDOPT and CEEWCOPT sample jobs that update installation-wide defaults (CICS and non-CICS) when new options are added, sub-options are added, or when options are deleted.

   Recommendation: Do one of the following:

- Take new copies of the CEEDOPT and CEECOPT members from CEE.SCEESAMP and modify them according to the procedures outlined in *z/OS Language Environment Customization*.
- Apply necessary changes to existing sample jobs after reviewing the summary of changes in *z/OS Language Environment Programming Reference* pertaining to run-time options.

Changes might be required to the CEEWROPT sample jobs that set region defaults for CICS and IMS/LRR environments. When options receiving new sub-options or when deleted options are specified in these sample jobs, changes are required.

Recommendation: Apply necessary changes to existing sample jobs after reviewing the summary of changes in *z/OS Language Environment Programming Reference* pertaining to run-time options. Specifically, look for options with new sub-options or deleted options.

Changes might be required to the CEEWUOPT sample jobs that set application defaults. Changes are necessary to existing sample jobs only if it is desired to use settings other than the defaults for those options where new sub-options have been added, or when deleted options are specified. Review the summary of changes in *z/OS Language Environment Programming Reference* pertaining to run-time options to determine if changes to these sample jobs are needed.

For the CEEWROPT and CEEWUOPT sample jobs, changes would only be necessary to the existing jobs if you run them again on the new release. Consider changing any or all sample jobs when option defaults are changed.

5. In z/OS V1R7, a new PARMLIB member, CEEPRMxx, was added for Language Environment. You can use it to specify Language Environment run-time options for the system. Operator commands are also provided to allow you to query and update the active run-time options for the system. This simplifies the management of Language Environment options, particularly in multisystem environments, and makes it possible to move Language Environment customization out of assembler language modules maintained using SMP/E usermods. However, specifying Language Environment options using CEEDOPT, CEECOPT and CELQDOPT modules continues to be supported. For more information, see *z/OS Language Environment Customization* or *z/OS Language Environment Programming Reference*.

Migration considerations for Language Environment in z/OS V1R13

There are no run-time application migration considerations for z/OS V1R13.

Migration considerations for Language Environment in z/OS V1R12

Setting run-time options as overrideable or nonoverrideable

Description: Before z/OS V1R12, all run-time options specified in a CEEPRMxx parmlib member were overrideable by default. Beginning with z/OS V1R12, you can set run-time options as overrideable or nonoverrideable in the CEEPRMxx parmlib member or with a SETCEE command using the OVR or NONOVR attribute. The ability to specify an option as overrideable or nonoverrideable removes a barrier to using CEEPRMxx.

Element or feature:	Language Environment
When change was introduced:	z/OS V1R12.
Applies to migration from:	z/OS V1R11 and z/OS V1R10.

Timing:	Before the first IPL of z/OS V1R12.
Is the migration action required?	No, but recommended so you can eliminate use of the assembler language USERMODs to specify installation-wide run-time options, and use parmlib member CEEPRMxx instead.
Target system hardware requirements:	None.
Target system software requirements:	None.
Other system (coexistence or fallback) requirements:	None.
Restrictions:	None.
System impacts:	None.
Related IBM Health Checker for z/OS check:	None.

Steps to take: Set run-time options as overrideable or nonoverrideable in the CEEPRMxx parmlib member or by issuing the SETCEE command using the OVR or NONOVR attribute.

Now that run-time options can be specified as overrideable or nonoverrideable in a CEEPRMxx parmlib member, and with a SETCEE command, you can eliminate the use of assembler language USERMODs to specify installation-wide run-time options.

Reference information:

- For more details about specifying a CEEPRMxx parmlib member, see *z/OS Language Environment Customization*.
- For the updated CEEPRM00 sample parmlib member, which includes every option specified as overrideable, see the CEE.SCEESAMP data set.

Changes to the Language Environment run-time options report when issuing D CEE

Description: Starting in z/OS V1R12, changes are made to the Language Environment run-time options report when a D CEE command is issued. Before z/OS V1R12, the Language Environment run-time options report displayed all suboptions, even if they were not explicitly set for any run-time option that was specified. Starting in z/OS V1R12, when a valid option is specified with a parmlib member or a SETCEE command, only the suboptions specified are displayed when a D CEE command is issued. A comma is displayed as a placeholder for those suboptions not specified.

Element or feature:	Language Environment
When change was introduced:	z/OS V1R12.
Applies to migration from:	z/OS V1R11 and z/OS V1R10.
Timing:	Before the first IPL of z/OS V1R12.
Is the migration action required?	Yes, if you have an application that reads the output of a D CEE command.
Target system hardware requirements:	None.
Target system software requirements:	None.

Other system (coexistence or fallback) requirements:	None.
Restrictions:	None.
System impacts:	None.
Related IBM Health Checker for z/OS check:	None.

Steps to take:

Examine any programs that read the output of a D CEE command to ensure compatibility with the updated run-time options report. Commas are now displayed for any suboptions that are not explicitly specified in a parmlib member or with a SETCEE command.

For example, if the following SETCEE command is issued:

```
SETCEE CEEDOPT,ALL31,ANYHEAP(4K),FILETAG(,AUTOTAG)
```

a subsequent D CEE,CEEDOPT command displays the following:

```
CEE3745I 09.32.13 DISPLAY CEEDOPT
NO MEMBERS SPECIFIED
LAST WHERE SET                          OPTION
------------------------------------------------------------------------
SETCEE command                           ALL31()
SETCEE command                           ANYHEAP(4096,,,)
SETCEE command                           FILETAG(,AUTOTAG)
```

Reference information: For more information, see *z/OS Language Environment Customization* and *z/OS MVS System Commands*.

Removal of conversion table source code

Description: Starting in z/OS V1R12, the C/C++ run-time library will no longer ship any ucmap source code or genxlt source code for character conversions now being performed by Unicode Services.

Element or feature:	Language Environment
When change was introduced:	z/OS V1R12.
Applies to migration from:	z/OS V1R11 and z/OS V1R10.
Timing:	After the first IPL of z/OS V1R12.
Is the migration action required?	Yes, if you use the iconv() family of functions to test to a "known conversion result" and experience testcase failures. Also, if you use custom conversion tables replacing those listed in either ucmapt.lst or genxlt.lst.
Target system hardware requirements:	None.
Target system software requirements:	None.
Other system (coexistence or fallback) requirements:	None.
Restrictions:	None.
System impacts:	None.
Related IBM Health Checker for z/OS check:	None.

Steps to take:

- If you use customized conversion tables, you should now generate custom Unicode Services conversion tables.
- If you use the iconv() family of functions testing to a "known conversion result" and experience test case failures, you need to update your expected results to the new conversion results.
- If you want to create custom conversion tables involving any of the CCSIDs related to the conversion table source no longer being shipped, you should now generate custom Unicode Services conversion tables instead of custom Language Environment conversion tables.

The *installation prefix*.SCEEUMAP data set will no longer be shipped.

The /usr/lib/nls/locale/ucmap HFS directory will no longer be shipped.

Note: The _ICONV_TECHNIQUE environment variable must be set to the same technique search order value used for the customized Unicode Services table in order for the iconv() family of functions to use the customized Unicode Services table. For example, if you want the iconv() family of functions to use a user-defined Unicode Services table with a technique search order of 2, the _ICONV_TECHNIQUE environment variable should be set to 2LMREC.

Reference information: For information on how to generate and use custom Unicode Services conversion tables, see *z/OS Unicode Services User's Guide and Reference*.

Changes to the CICS CLER run-time options report

Description: Starting with z/OS V1R12, the Language Environment run-time options report displayed from the CICS CLER transaction is changed. The report is modified to have a wider LAST WHERE SET column to accomodate longer values, such as "Installation Non-overrideable." In addition, the report heading OPTIONS is changed to OPTION to match the other run-time options reports.

Element or feature:	Language Environment.
When change was introduced:	z/OS V1R12.
Applies to migration from:	z/OS V1R11 and z/OS V1R10.
Timing:	After the first IPL of z/OS V1R12.
Is the migration action required?	Yes, if you have an application that reads the output of a CICS CLER transaction.
Target system hardware requirements:	None.
Target system software requirements:	None.
Other system (coexistence or fallback) requirements:	None.
Restrictions:	None.
System impacts:	None.
Related IBM Health Checker for z/OS check:	None.

Steps to take: Examine programs that read the output of a CICS CLER transaction to ensure compatibility with the updated CLER run-time options report. The LAST

WHERE SET column is now wider and the OPTIONS heading is changed to OPTION. The following is a subset of the new report to show the formatting changes:

```
LAST WHERE SET                OPTION
----------------------------  ------------------------------------------------
Installation default            ABPERC(NONE)
Installation default            ABTERMENC(ABEND)
Installation default          NOAIXBLD
```

Reference information: For more information, see *z/OS Language Environment Programming Guide* and *z/OS Language Environment Debugging Guide*.

Migration considerations for Language Environment in z/OS V1R11

Changes to the HEAPCHK run-time option

The HEAPCHK run-time option now has 4 additional sub-options. Users who use the CEEDOPT, CEECOPT and CELQDOPT usermods to set their installation default run-time options must make a change. Users who use CEEPRMxx to set their system default run-time options rather than the user mods are unaffected.

Steps to take if you use the usermods:

1. Consider using the CEEPRMxx parmlib member.
2. Compare your existing source for the installation-wide run-time options CSECT, CEEDOPT (non-CICS environment), CEECOPT (CICS environment), or CELQDOPT (AMODE 64 environment) with the new samples in the hlq.SCEESAMP data set to determine whether you need to change your defaults. If changes are necessary, you must make them and apply the corresponding usermods.
3. Understand the new HEAPCHK run-time option sub-options and their default values.
4. Determine if the default values are acceptable for your installation and adjust if needed.

Reference information: For details about specifying the HEAPCHK run-time option, see *z/OS Language Environment Customization* or *z/OS Language Environment Programming Reference*

Changes to Binary and Decimal Floating-Point support in the CICS Environment

Certain Binary and Decimal Floating-Point Exceptions that were previously reported with a CEE3207S message are now reported with the following messages: CEE3216S, CEE3217S, CEE3218S, CEE3219S, CEE3220S, CEE3221S, CEE3222S, CEE3223S, CEE3224S, CEE3225S, CEE3226S, CEE3227S, CEE3228S, CEE3229S, CEE3231S, CEE3232S, or CEE3233S. These messages are the same ones uses in non-CICS environments for Floating-Point exceptions.

The floating-point control register (FPC), floating-point registers 1,3,5,7,8-15, access registers (ARs), and high registers (HRs) are now saved and restored when applications are resumed after program checks and ABENDs when CICS TS 4 or higher is used. Any changes to these registers made by user condition handlers or signal catchers may be ignored when the registers are restored and the application is resumed.

The sample USRHDLR program, CEEWUCHA, has been changed to check for the new floating program check conditions 3216-3229 and 3231-3233. Any customized versions of CEEWUCHA that are in use may need to be updated.

Migration considerations for Language Environment in z/OS V1R10

HEAPPOOLS Consideration:

HEAPPOOLS was ignored for AMODE64 applications in previous releases, when specified using _CEE_RUNOPTS, but is supported as of z/OS V1R10. In previous releases, when an AMODE64 application spawned an AMODE31 process, and the HEAPPOOLS run-time option was specified using _CEE_RUNOPTS, it would be ignored by the AMODE64 application, but would be propogated and accepted by the AMODE31 process. Additionally, when specified using _CEE_RUNOPTS, if an AMODE31 application spawned an AMODE64 process, the AMODE31 application would accept the HEAPPOOLS run-time option, but the AMODE64 process would ignore it.

The HEAPPOOLS run-time option is no longer ignored when specified from _CEE_RUNOPTS in AMODE 64 applications. Users who format the CEEOCB must now format the HEAPPOOLS run-time option for AMODE 64 environments.

This support is also available to the following releases:

- z/OS V1R7 with PK41618
- z/OS V1R8 with PK47298
- z/OS V1R9 with PK49427

Note: PK44554 adds storage report and heap pools trace support for the above releases.

Health Check - check(ibmcee,cee_using_le_parmlib):

This check verifies the use of LE parmlib support. See *IBM Health Checker for z/OS: User's Guide* for more information.

Changes to CEEDOPT location:

The CSECT CEEDOPT run-time option is located in CEEPLPKA. If you do not manually link the run-time option you are not affected by this location change.

Note: If you use the ++USERMOD(CEEWD01) to update the installation-wide run-time options, you are not affected by this location change.

Changes to data exceptions:

Data exceptions generated by the Compare and Trap family of instructions (such as CRT, CGRT, CGFRT, CIT, CGIT, CLRT, CLGRT, CLGFRT, CLFIT, and CLGIT) will be handled differently in V1R10. These data exceptions are now result in message CEE3234S instead of CEE3207S. Also, if the condition results in a POSIX signal, the signal that is generated is SIGFPE with a signal code (si_code) of FPE_CTDXC.

Changes to PL/I stream I/O behavior:

PL/I stream I/O behavior has been changed in some situations. See Chapter 5, "Other HLL migration considerations," on page 27 for more information.

Changes to Venezuela and Malta currency:

The currency in Venezuela has changed from bolivar to bolivar fuerte. The national currency symbol has changed from Bs to BSF and the international currency symbol has changed from VEB to VEF. C/C++ users who want to keep using the old currency symbols, the Bs or VEB (bolivar), must use setlocale() with a locale name of "Es_VEO" for the language-territory part, instead of "Es_VE". Users of the CEE3MCS and CEE3MC2 callable services can only access the new currency symbols.

Malta is adopting the euro currency. Users who want to keep using the old currency symbol must use the @preeuro locales.

Migration considerations for Language Environment in z/OS V1R9

CEEDUMP run-time option added

The CEEDUMP run-time option is used to specify options to control the processing of the Language Environment dump report known as CEEDUMP.

Among the possible installation-wide options that can be used to affect the Language Environment dumps are the following:

- Set the number of lines to be displayed on each page of the CEEDUMP report.
- Indicate the default SYSOUT class and form-name to use for all dynamically allocated CEEDUMPs.
- Set the default values for the FREE and SPIN JCL DD attributes to use for all dynamically allocated CEEDUMPs.

Steps to take:

1. Consider using the CEEPRMxx parmlib member.
2. Compare your existing source for the installation-wide run-time options CSECT, CEEDOPT (non-CICS environment), or CEECOPT (CICS environment) with the new samples in the hlq.SCEESAMP data set to determine whether you need to change your defaults. If changes are necessary, you must make them and apply the corresponding usermods.
3. Understand the new CEEDUMP run-time option and its default suboption values.
4. Determine if the default values are acceptable for your installation and adjust if needed.

Reference information: For details about specifying the CEEDUMP run-time option, see *z/OS Language Environment Customization* or *z/OS Language Environment Programming Reference*

Changes to Language Environment dump output, options reports and storage reports

Although none of these reports are programming interfaces, the following has changed:

- In the Language Environment dump output, the traceback now has multiple parts. New columns of data have been added and others have been re-arranged. Refer to the *z/OS Language Environment Debugging Guide* for examples of the new format.

- In the storage report, the HeapPools Summary section has a new column called Element Size and the column called Cell Size is now called Specified Cell Size. Refer to the *z/OS Language Environment Debugging Guide* for examples of the new format.
- Language Environment dump output, options reports and storage reports are now sensitive to the national language. When the national language is uppercase U.S. English or Japanese, titles, heading and keywords that were in mixed case U.S. English will now be in upper case U.S English.

Changes to CELQPIPI

Any existing AMODE 64 PIPI application that currently specifies a non-zero value in the reserved third parameter of the CELQPIPI INIT_MAIN or INIT_SUB function must specify 0 in this 3rd parameter or it must point to an AMODE 64 PIPI service routine vector.

For more information, refer to the *z/OS Language Environment Programming Guide for 64-bit Virtual Addressing Mode*.

Changes to the CLER transaction

The CLER (CICS transaction) panel has changed as follows:

- The order of options on the CLER panel has changed to show them alphabetically.
- CHECK and INFOMSGFILTER have been added to the list of options.
- CLER users will now be prompted for confirmation if you use:
  - The ON suboption of CBLPSHPOP, RPTSTG and RTPOPT
  - The OFF suboption of ALL31

There are no actions you need to take.

Changes to XPLINK under LRR

z/OS V1.9 Language Environment now supports running XPLINK applications in an LRR (Library Routine Retention) environment. The new support requires the user to indicate if the LRR environment can support XPLINK applications. The default behavior when setting up an LRR environment is that it does not support XPLINK applications. If an attempt is made to run an XPLINK application in an LRR environment that does not support XPLINK, an ABEND will result.

There are two minor migration issues associated with this new support:

- Prior to z/OS V1.9, when a non-XPLINK main with XPLINK(ON) run-time option was run in an LRR environment, Language Environment forced the XPLINK run-time option to OFF and allowed the application to continue. For z/OS V1.9, this same application, when run in an LRR environment that does not support XPLINK, causes Language Environment to issue ABEND U4093 with reason code 178 (X'B2').
- Prior to z/OS V1.9, when an XPLINK main was run in an LRR environment, Language Environment issued ABEND U4093 with reason code 176 (X'B0'). For z/OS V1.9, this same application, when run in an LRR environment that does not support XPLINK, causes Language Environment to issue ABEND U4093 with reason code 177 (X'B1').

Heap Pools Consideration:

HEAPPOOLS and HEAPPOOLS64 changes

The storage required for HEAPPOOLS control structures has increased by approximately 3k.

The storage required for HEAPPOOLS64 control structures has increased by approximately 3k.

Note: This change does not affect user heap pools created with the __ucreate() function.

Changes to messages

- The following messages have been added to Language Environment for this release as part of the support for IEEE Decimal Floating Point (DFP) for C/C++ programs:
  - EDC6259S
  - CEE3226S
  - CEE3227S
  - CEE3228S
  - CEE3229S
  - CEE3231S
  - CEE3232S
  - CEE3233S
- The following messages have been updated for this release as part of the DLL enhancements for C/C++ programs:
  - EDC5204E
  - EDC5205S
  - EDC5206S
  - EDC5207S
  - EDC5208I
  - EDC5209I
  - EDC5210I
  - EDC5212I
  - EDC5213I
  - EDC5214I
  - EDC5215I
  - EDC5216I
  - EDC5217I
  - EDC5218I
  - EDC5220I
  - EDC5221S
  - EDC5225E
  - EDC5226S
  - EDC5234S
  - EDC5237S
  - EDC5251I
  - EDC5252S
  - EDC5253S
  - EDC5254S

Chapter 3. Migrating from other run-time environments

This topic describes, in general, the compatibility of Language Environment with previous run-time libraries. It also describes what you must do to migrate different object and load modules to Language Environment.

Note: This publication does not describe all migration considerations. For a detailed description of migration considerations, see the appropriate language migration guide listed in the following topic.

Compatibility with previous run-time libraries

With certain exceptions, Language Environment provides object and load module compatibility for applications that are generated with the following pre-Language Environment IBM language products. Load modules that are created with these compilers and link-edited with their associated run-time libraries run compatibly with Language Environment without relinking. Also, object modules created with these compilers can be linked and run with Language Environment without recompiling.

- C/370 Versions 1 and 2
- OS/VS COBOL Release 2
- VS COBOL II Release 3 or later
- OS PL/I Version 1 Release 3 (object modules), Version 1 Release 5.1 and Version 2, all releases (load modules)
- VS FORTRAN Versions 1 and 2 (MVS only)
- FORTRAN IV H Extended (MVS only)
- FORTRAN IV G1 (MVS only)

The following topics contain some basic information to help you determine if your applications will run compatibly with Language Environment. For more detailed information about compatibility, see one of the following migration guides:

- *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*
- *IBM C for VM/ESA Compiler and Run-Time Migration Guide*
- *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409.
- *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide*
- *PL/I for MVS & VM Compiler and Run-Time Migration Guide* or *Enterprise PL/I for z/OS, V3R9, Migration Guide*
- *Fortran Run-Time Migration Guide*

Migrating ILC applications to Language Environment

Table 2 lists some of the compatibility exceptions you should consider when migrating ILC applications to Language Environment.

Table 2. ILC Compatibility Exceptions

To Migrate:	You Need To:
Load modules that contain OS/VS COBOL, with calls to, or from, OS PL/I	Upgrade the COBOL source code and compile with Enterprise COBOL for z/OS or COBOL for OS/390 & VM.
Load modules that contain VS COBOL II Version 1 Release 3 or later, with calls to, or from, OS PL/I	Relink with Language Environment. However, if you link your VS COBOL II-OS PL/I ILC applications with the migration tool provided by OS PL/I Version 2 Release 3, you will not need to relink your applications. The PTF numbers for the migration aid are UN76954 and UN76955. For information about the migration tool, see *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide*.
C/370 Version 2 Release 2 (V2R2) load modules that contain calls to, or from, VS COBOL II, COBOL/370, or COBOL for MVS & VM programs	Apply the PTF associated with APAR PN74931, which allows you to relink C/370 V2R2 load modules with the C/370 V2R2 library and run with Language Environment or the C/370 V2R2 library.
Load modules that contain Fortran with calls to, or from, any other language	Relink the load modules with z/OS Language Environment, using the Language Environment libraries rather than pre-Language Environment Fortran libraries. Fortran and PL/I provide migration tools. For information about the Fortran library replacement tool, see *z/OS Language Environment Programming Guide*; for information about the PL/I migration tool, see *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide*.

See *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*, or *IBM C for VM/ESA Compiler and Run-Time Migration Guide* for detailed instructions on how to relink C-COBOL ILC applications. (You do not need to relink PL/I-C ILC applications.)

See *Enterprise PL/I for z/OS, V3R9, Migration Guide*, *PL/I for MVS & VM Compiler and Run-Time Migration Guide* or *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide* for instructions on how to relink PL/I-COBOL ILC applications, and for information about a migration aid that helps migrate OS PL/I-VS COBOL II ILC applications.

For more information about relinking C-Fortran ILC applications, see *z/OS Language Environment Programming Guide* or *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*.

Migrating C routines to Language Environment

Generally, you can directly migrate most C/370 Version 1 or Version 2 applications to any release of Language Environment. However, you must use the Language Environment libraries to relink an application if a load module contains one of the following items:

- ILC calls to, and from, Fortran or in some cases COBOL (see Table 2 on page 16)
- Debugging information (that is, they are compiled with the TEST option)
- System Programming C Facility (SPC) load modules that contain dynamic C/370 library functions

For detailed information about migrating your C applications, see *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*.

Migrating COBOL programs to Language Environment

Table 3 contains a subset of COBOL compatibility exceptions.

Table 3. COBOL Compatibility Exceptions

To Migrate:	You Need To:
OS/VS COBOL programs mixed with assembler under non-CICS	Run in a single run unit (SVC LINK is not allowed).
OS/VS COBOL programs that use ILC with PL/I	Upgrade the COBOL source code to Enterprise COBOL for z/OS or COBOL for OS/390 & VM.
OS/VS COBOL programs that use ILC with FORTRAN	Upgrade the COBOL source code to Enterprise COBOL for z/OS or COBOL for OS/390 & VM.
VS COBOL II programs that use ILC with C or PL/I	See Table 2 on page 16 for information about migration aids for each language.

Recommendation: You should not install more than one library for a language in the LNKLST or LPALST. You should not concatenate pre-Language Environment run-time libraries in the LNKLST.

See *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527 and *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409 for detailed migration information about LNKLST concatenation and COBOL.

Migrating Fortran routines to Language Environment

Table 4 lists some compatibility exceptions to consider when migrating Fortran applications to Language Environment. For more information, see *Fortran Run-Time Migration Guide*.

Table 4. Fortran Compatibility Exceptions

To Migrate:	You Need To:
Object modules compiled with VS FORTRAN Version 1 Release 2.0 or earlier and are either programs or subprograms that receive character arguments or pass character arguments to subprograms	Recompile with VS FORTRAN Version 2 and run under Language Environment.

Table 4. Fortran Compatibility Exceptions (continued)

To Migrate:	You Need To:
Object modules compiled with VS FORTRAN Version 2 Release 5 or 6 that contain parallel constructs, use the PARALLEL compile-time option, or invoke PEORIG, PEPOST, PEWAIT, PETERM, PLCOND, PLFREE, PLLOCK, PLORIG, or PLTERM	Continue to link-edit and run under VS FORTRAN Version 2. These object modules cannot run under Language Environment.
Object modules compiled with VS FORTRAN Version 2 Release 5 or 6 using the EC compiler option	Perform one of the following actions, as these object modules cannot run under Language Environment: • Continue to link-edit and run under VS FORTRAN Version 2 Release 5 or 6, or • Remove the EC option from your source, if possible, then recompile and run with Language Environment.
Object modules with calls to DVCHK or OVERFL services	Remove the calls, change the logic of the program and recompile with VS FORTRAN Version 2.
Object modules that have dependences on product internals	Remove the dependencies, change the logic of the program and recompile with VS FORTRAN Version 2.
Object modules that have misaligned vector operands	Ensure that all vector operands are properly aligned and recompile with VS FORTRAN Version 2.
Object modules that use static debug	Remove the debug packets and recompile with VS FORTRAN Version 2.
Load modules that contain Fortran with calls to, or from, any other language	See Table 2 on page 16 for instructions.

Migrating PL/I routines to Language Environment

Table 5 lists some compatibility exceptions for migrating PL/I routines to Language Environment. See *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide*, *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide* for more information.

Table 5. PL/I Compatibility Exceptions

To Migrate:	You Need To:
Object modules created with OS PL/I Version 1 Release 1 through Version 1 Release 2.3 compilers	Recompile with Enterprise PL/I, PL/I for MVS & VM or with the OS PL/I Version 2 compiler.
Load modules created with OS PL/I Version 1 Releases 3 through 5.	Relink with Language Environment or with OS PL/I Version 2.
Load modules created with OS PL/I Version 1 Release 5.1.	Apply the IBM-supplied program fix (ZAP) before running the following types of OS PL/I V1 R5.1 load modules: • Main load modules for MVS non-shared library, non-CICS, nonmultitasking

Table 5. PL/I Compatibility Exceptions (continued)

To Migrate:	You Need To:
Load modules that use the OS PL/I shared library	Relink or recompile load modules from OS PL/I Version 1 Releases 1 through 5 shared library; these load modules are not supported. Load modules from OS PL/I Version 1 Release 5.1 and the Version 2 shared library are supported; however, you must rebuild the shared library once under Language Environment.

Migrating assembler programs to Language Environment

To run assembler programs with Language Environment, you must ensure the assembler programs adhere to conventions for items such as register and storage usage, condition handling, and accessing input parameters. For example, assembler programs must set a valid 31 bit address in the save area back chain.

Language Environment provides several assembler macros, which your assembler programs should use to perform tasks such as entering and exiting assembler routines and mapping Language Environment data areas. For example, when you use the CEEENTRY and CEETERM macros, Language Environment automatically initializes and terminates, respectively, the execution environment for the application. In addition, when the Language Environment environment is established for the main assembler program, that environment is also established for any other routines that may be called later.

For more information about assembler considerations and Language Environment macros, see *z/OS Language Environment Programming Guide*. For more information about assembler considerations when assembler programs are used with COBOL, see *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409.

Chapter 4. Choosing run-time options for compatible behavior

This topic provides information on how Language Environment run-time options differ from run-time options that are specific to a high-level language (HLL). For more information about run-time options, see the following publications:

- *z/OS Language Environment Customization*
- *z/OS Language Environment Programming Reference*

Differences between run-time options

Language Environment provides a set of run-time options for applications. These options are processed at the enclave level and allow you to control many aspects of the Language Environment environment. The options comparison tables show how Language Environment run-time options differ from the run-time options that are specific to C, COBOL, Fortran, and PL/I (if an HLL run-time option is not listed in a table, you can assume it operates under Language Environment in the same way it did before Language Environment):

High-Level Language	Language Environment Option Information
C	Table 6
COBOL	Table 7 on page 22
Fortran	Table 8 on page 24
PL/I	Table 9 on page 25

C and Language Environment run-time options comparison

Table 6. C and Language Environment Run-Time Options

C Option	Language Environment Equivalent	Notes
ISAINC	STACK	If you do not change the C/370 run-time option ISAINC, you will receive a warning message during execution.
ISASIZE	STACK	If you do not change the C/370 run-time option ISASIZE, you will receive a warning message during execution.
LANGUAGE	NATLANG	Mixed-case and uppercase US English and Japanese are supported. If you do not change the C/370 run-time option LANGUAGE, you will receive a warning message during execution.
REPORT | NOREPORT	RPTSTG(ON | OFF), RPTOPT(ON | OFF)	RPTSTG(ON | OFF) and RPTOPT(ON | OFF) provide behavior compatible with REPORT | NOREPORT, and affect all languages in an enclave. If you do not change the C/370 run-time option REPORT | NOREPORT, you will receive a warning message during execution.
SPIE | NOSPIE STAE | NOSTAE	TRAP(ON,SPIE) TRAP(OFF)	If either SPIE or STAE is specified or defaulted in input, TRAP is set to TRAP(ON,SPIE). If both NOSPIE and NOSTAE are specified, TRAP is set to TRAP(OFF). TRAP(ON,SPIE) is the recommended setting.

COBOL and Language Environment run-time options comparison

Table 7. COBOL and Language Environment Run-Time Options

COBOL Option	Language Environment Equivalent	Notes
AIXBLD | NOAIXBLD	AIXBLD | NOAIXBLD	Invokes the access methods services for VSAM indexed and relative data sets to complete the file and index definition procedures for COBOL routines. Under z/OS, Access Method Services (AMS) messages are directed to the ddname specified in the Language Environment run-time option MSGFILE. Under CMS, the messages are erased, which is the same behavior as VS COBOL II. AIXBLD | NOAIXBLD is not applicable under CICS.
DEBUG | NODEBUG	DEBUG | NODEBUG	DEBUG | NODEBUG provides behavior compatible with VS COBOL II.
FLOW | NOFLOW	FLOW | NOFLOW	FLOW | NOFLOW provides behavior compatible with VS COBOL II.
LANGUAGE	NATLANG	NATLANG replaces LANGUAGE, which is a VS COBOL II installation option. You can select a national language at run time or installation time by using the NATLANG option.
LIBKEEP | NOLIBKEEP	Not applicable	LIBKEEP | NOLIBKEEP is not supported under Language Environment and is not applicable under CICS. To obtain similar function, use the Library Routine Retention (LRR) feature, which is described in *z/OS Language Environment Programming Guide*. To use LRR in an IMS/TM environment, see *z/OS Language Environment Customization*.
MIXRES | NOMIXRES	Not applicable	MIXRES | NOMIXRES is not supported under Language Environment and is not applicable under CICS. Mixed RES and NORES applications when linked with Language Environment will exhibit RES-like behavior; see *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409 for more information.
QUEUE	Not applicable	QUEUE is not supported under Language Environment.

Table 7. COBOL and Language Environment Run-Time Options (continued)

COBOL Option	Language Environment Equivalent	Notes
RTEREUS | NORTEREUS	RTEREUS | NORTEREUS	RTEREUS is not recommended as an installation default. Use RTEREUS only for specific applications and ensure that you understand the possible side effects, for example: • Under Language Environment, RTEREUS(ON) is only supported in a single enclave environment. Applications that create multiple enclaves will terminate with error message IGZ0168S. Multiple enclaves can be created by applications that use SVC LINK or CMSCALL to invoke application programs. One example is when an SVC LINK is used to invoke an application program under ISPF that is using ISPF services (such as CALL 'ISPLINK' and ISPF SELECT). • If a Language Environment reusable environment is established (using RTEREUS), attempts to run a C or PL/I main program under Language Environment will fail. For example, when running on ISPF with RTEREUS(ON): – The first program invoked by ISPF is a COBOL program; a Language Environment reusable environment is established. – At another point, ISPF invokes a PL/I or C program; the initialization of the PL/I or C program will fail. • If many COBOL programs are run under the same z/OS task, "out of storage" abends may occur. This occurs because all storage acquired by Language Environment to run COBOL programs is kept until the z/OS task ends or the Language Environment environment terminates. • Language Environment does not produce storage and run-time options reports unless STOP RUN is issued to end the enclave.
SIMVRD | NOSIMVRD	SIMVRD | NOSIMVRD	SIMVRD | NOSIMVRD provides behavior compatible with the VS COBOL II SIMVRD | NOSIMVRD option.
SPOUT | NOSPOUT	RPTOPTS(ON | OFF), RPTSTG(ON | OFF)	Storage reports are directed to the ddname specified in the Language Environment option MSGFILE. For more information, see *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409.
SSRANGE | NOSSRANGE	CHECK(ON | OFF)	CHECK(ON | OFF) provides behavior compatible with SSRANGE | NOSSRANGE.
STAE | NOSTAE	TRAP(ON,SPIE) TRAP(OFF)	If STAE | NOSTAE is specified in input, then TRAP is set according to the option: TRAP(ON,SPIE) for STAE, and TRAP(OFF) for NOSTAE. TRAP(ON,SPIE) is the recommended setting.
UPSI	UPSI	UPSI provides behavior compatible with the VS COBOL II UPSI option.
WSCLEAR | NOWSCLEAR	STORAGE(00,,,)	For behavior similar to WSCLEAR | NOWSCLEAR, use the Language Environment STORAGE(00,,,) option. For more information, see *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409.

Fortran and Language Environment run-time options comparison

Table 8. Fortran and Language Environment Run-Time Options

Fortran Option	Language Environment Equivalent	Notes
ABSDUMP | NOABSDUMP	TERMTHDACT	TERMTHDACT(DUMP) replaces ABSDUMP to produce a Language Environment dump at termination, but there is no automatic mapping. TERMTHDACT with suboptions TRACE, QUIET, MSG, UATRACE, UAONLY, or UAIMM replaces NOABSDUMP to avoid getting a Language Environment dump at termination.
AUTOTASK | NOAUTOTASK	AUTOTASK | NOAUTOTASK	AUTOTASK | NOAUTOTASK provides behavior compatible with VS FORTRAN Version 2.
CNVIOERR | NOCNVIOERR	Not applicable	There is no Language Environment equivalent for CNVIOERR | NOCNVIOERR. Fortran semantics are as though CNVIOERR were in effect.
DEBUG | NODEBUG	Not applicable	There is no debugger support for Fortran.
DEBUNIT	Not applicable	There is no Language Environment equivalent for DEBUNIT.
ECPACK | NOECPACK	Not applicable	There is no Language Environment equivalent for ECPACK | NOECPACK. You cannot run programs with Language Environment that use access registers or that were compiled with the EC or EMODE compiler options.
ERRUNIT	ERRUNIT	ERRUNIT provides behavior compatible with VS FORTRAN Version 2.
FAIL	ABTERMENC	ABTERMENC replaces FAIL, but there is no automatic mapping. ABTERMENC controls whether a condition of severity 2 or greater is terminated with a return code or an abend. ABTERMENC(RETCODE) is similar to FAIL(RC), and ABTERMENC(ABEND) is similar to FAIL(ABEND).
FILEHIST | NOFILEHIST	FILEHIST | NOFILEHIST	FILEHIST | NOFILEHIST provides behavior compatible with VS FORTRAN Version 2.
INQPCOPN | NOINQPCOPN	INQPCOPN | NOINQPCOPN	INQPCOPN | NOINQPCOPN provides behavior compatible with VS FORTRAN Version 2.
IOINIT | NOIOINIT	Not applicable	There is no Language Environment equivalent for IOINIT | NOIOINIT. The message file is opened either when the first record is written to it or when an OPEN statement refers to error message unit. If no allocation for the ddname has been made for the message file, it is dynamically allocated to the terminal (under TSO) or to SYSOUT=* (under z/OS batch).

Table 8. Fortran and Language Environment Run-Time Options (continued)

Fortran Option	Language Environment Equivalent	Notes
OCSTATUS | NOOCSTATUS	OCSTATUS | NOOCSTATUS	OCSTATUS | NOOCSTATUS provides behavior compatible with VS FORTRAN Version 2.
PARALLEL | NOPARALLEL	Not applicable	There is no Language Environment equivalent for PARALLEL | NOPARALLEL. Parallel programs cannot be run with Language Environment.
PC | NOPC	PC | NOPC	PC specifies that Fortran static common blocks with the same name but in different load modules do not refer to the same storage.
PRTUNIT	PRTUNIT	PRTUNIT provides behavior compatible with VS FORTRAN Version 2.
PTRACE | NOPTRACE	Not applicable	There is no Language Environment equivalent for PTRACE | NOPTRACE. Parallel programs cannot be run with Language Environment.
PUNUNIT	PUNUNIT	PUNUNIT provides behavior compatible with VS FORTRAN Version 2.
RDRUNIT	RDRUNIT	RDRUNIT provides behavior compatible with VS FORTRAN Version 2.
RECPAD | NORECPAD	RECPAD(OFF | NONE | VAR | ALL | ON)	NORECPAD automatically maps to RECPAD(OFF). Fortran does not support RECPAD(VAR). RECPAD must be changed to RECPAD(ON).
SPIE | NOSPIE STAE | NOSTAE	TRAP(ON,SPIE) TRAP(OFF)	If either SPIE or STAE is specified or defaulted in input, TRAP is set to TRAP(ON,SPIE). If both NOSPIE and NOSTAE are specified, TRAP is set to TRAP(OFF). TRAP(ON,SPIE) is the recommended setting.
XUFLOW | NOXUFLOW	XUFLOW(ON | AUTO) XUFLOW(OFF)	There is no automatic mapping of XUFLOW to the Language Environment XUFLOW. NOXUFLOW maps to the Language Environment XUFLOW(OFF), which provides compatible behavior.

PL/I and Language Environment run-time options comparison

Table 9. PL/I and Language Environment Run-Time Options

PL/I Option	Language Environment Equivalent	Notes
COUNT | NOCOUNT	Not applicable	There is no Language Environment equivalent for COUNT | NOCOUNT. It is not processed but produces an informational message.

Table 9. PL/I and Language Environment Run-Time Options (continued)

PL/I Option	Language Environment Equivalent	Notes
FLOW | NOFLOW	Not applicable	There is no Language Environment equivalent for FLOW | NOFLOW. Language Environment honors this option only as a COBOL option.
ISAINC	STACK, THREADSTACK, or PLITASKCOUNT	ISAINC maps to three Language Environment options, STACK, NONIPTSTACK, and PLITASKCOUNT, which provide compatible behavior.
ISASIZE	STACK, THREADSTACK, or PLITASKCOUNT	ISASIZE maps to three Language Environment options, STACK, NONIPTSTACK, and PLITASKCOUNT, which provide compatible behavior.
LANGUAGE	NATLANG	Mixed-case and uppercase U.S. English and Japanese are supported.
REPORT | NOREPORT	RPTSTG(ON | OFF) RPTOPTS(ON | OFF)	RPTSTG(ON | OFF) and RPTOPTS(ON | OFF) provide behavior compatible with REPORT | NOREPORT.
SPIE | NOSPIE STAE | NOSTAE	TRAP(ON,SPIE) TRAP(OFF)	If either SPIE or STAE is specified or defaulted in input, TRAP is set to TRAP(ON,SPIE). If both NOSPIE and NOSTAE are specified, TRAP is set to TRAP(OFF). TRAP(ON,SPIE) is the recommended setting.
TASKHEAP	THREADHEAP	THREADHEAP provides behavior compatible with TASKHEAP.

Chapter 5. Other HLL migration considerations

This topic includes migration concerns that are language-specific, such as: differences in how Language Environment and an HLL handle return codes, run-time messages, entry files, and user exits. For more information about these considerations, see the following informations:

- *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*
- *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527, or *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409
- *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide*
- *z/OS Language Environment Programming Guide*

C considerations

The following topics list some sample migration problems. For a complete list of migration considerations, see one of the C migration guides listed in the preceding topic.

Standard streams

Under z/OS Language Environment there is no longer an automatic association of ddnames SYSTERM, SYSERR, SYSPRINT with `stderr`. Command line redirection of the type 1>&2 is necessary in batch to cause `stderr` and `stdout` to share a device.

In C/370 Version 1 and Version 2, you could override the destination of error messages by redirecting `stderr`. Language Environment determines the destination of all messages from the new MSGFILE run-time option. For more information about the MSGFILE option, see *z/OS Language Environment Programming Guide*.

Passing command line parameters

In C/370 Version 1 or Version 2, if an error was detected with the parameters being passed to the main program, the program terminated with a return code of 8 and a message indicating the reason the program terminated. For example, if there was an error in the redirection parameters, the message would indicate that the program had terminated because of a redirection error. Under Language Environment, the same message is displayed, but the program also terminates with a 4093 abend, reason code 52 (X'34'). For more information about reason codes, see *z/OS Language Environment Debugging Guide*.

User exits

If CEEBXITA and IBMBXITA are present in a relinked C/370 Version 1 or Version 2 module, CEEBXITA will take precedence over IBMBXITA.

Time functions

If you are migrating from IBM C/370 (Version 1 or Version 2) or AD/Cycle C/370 (Version 1 Release 1 or Version 1 Release 2), you should be aware of the following change in time functions.

- The `ctime()`, `localtime()`, and `mktime()` functions will return Coordinated Universal Time (UTC) time unless customized locale information is available. When you customize the locale, time functions preserve the time and date and correctly adjust for daylight time on a given date. See *z/OS XL C/C++ Programming Guide* for more information about environment variables and customizing locale information.
- In POSIX and non-POSIX applications, you can use the `TZ` environment variable to supply the necessary time zone information for your location. Previously, for non-POSIX applications, you could supply customized locale information only by setting time zone and daylight information in the `LC_TOD` locale category.

Load modules that invoke a Debugging Tool

C/370 library application load modules that use `ctest()` to invoke the Debug Tool must be relinked to run with Language Environment. The old library object, @@CTEST, must be replaced, as described in *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer* and *IBM C for VM/ESA Compiler and Run-Time Migration Guide*. After you replace the old objects, the new modules will run with Language Environment.

Prefix of perror() and strerror() messages in C

With Language Environment, all `perror()` and `strerror()` messages in C contain a prefix. With C/370 Version 1 and Version 2 there was no prefix on these messages. The prefix is EDC*xxxxa*, where *xxxx* is a number (always *5xxx*) and the *a* is I, W, or E. See *z/OS Language Environment Run-Time Messages* for a list of messages.

AMODE errors from ILCs

In ILC applications of C/370 Version 1 or Version 2 and VS COBOL II, if C/370 was running at AMODE 31 and COBOL was running at AMODE 24, an error was produced (2052) and the application failed. Under Language Environment, the call will fail but the message will be `EDC5052`, protection exception.

PL/I considerations

The following topics describe some of the items you should consider when migrating PL/I applications to Language Environment.

Dumps

The output produced by PLIDUMP is different when running under Language Environment. For detailed information, see *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide*, *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide*.

Condition handling

In general, PL/I condition handling functions in the same way when running under Language Environment. However, the issuing of diagnostic messages may vary. For example, the diagnostic message for an ERROR condition is issued only if there is no ERROR ON-unit established, or if the ERROR ON-unit does not recover from the condition by using a GOTO out of block. However, for other PL/I conditions whose implicit action includes printing a message and raising the ERROR condition, the message is issued before control is given to an established ERROR ON-unit.

User exits

The OS PL/I Version 2 assembler user exits IBMBXITA and IBMFXITA are supported by PL/I for MVS & VM for compatibility. However, the Language Environment user exit CEEBINT should be used instead. Only CEEBINT is supported by VisualAge PL/I for OS/390.

Also, the OS PL/I Version 2 high-level language user exit IBMBINT is not recommended; it is supported only for compatibility. Use the Language Environment high-level language user exit, CEEBINT, instead. Only CEEBXITA is supported by VisualAge PL/I for OS/390. See *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide*, *Enterprise PL/I for z/OS, V3R9, Migration Guide* or *PL/I for MVS & VM Compiler and Run-Time Migration Guide* for detailed information. See *z/OS Language Environment Programming Guide* for more information about the Language Environment user exits.

SYSPRINT

In PL/I, run-time messages are directed, by default, to the Language Environment MSGFILE rather than to SYSPRINT. Run-time user output is still directed to SYSPRINT. If you want run-time messages to go to SYSPRINT, specify the MSGFILE(SYSPRINT) run-time option. In this case, SYSPRINT can contain both user output and run-time output. For more information about the MSGFILE run-time option, see *z/OS Language Environment Programming Reference*.

If you specify a RECSIZE value that is not consistent with the LRECL of the data set or with the LRECL on the DD statement, the PL/I run-time library will diagnose this with an UNDEFINEDFILE condition with ONCODE=81. You must change the JCL to ensure that the values are the same or remove the LRECL value from the DD statement.

For DB2 UDB for z/OS Version 8 and DB2 Version 9.1 for z/OS customers, job steps that execute program DSNTEP2 or DSNTEP4 will experience user abend 4038. The user abend 4038 happens because UNDEFINEDFILE condition with *ONCODE=81* error when the SYSPRINT DD specifies an LRECL that does not match the RECSIZE specified by DSNTEP2 and DSNTEP4 in the PAGEWIDTH constant. PAGEWIDTH is typically 133 but can be changed in the source code for DSNTEP2 and DSNTEP4. If you experience the abend and do not know the PAGEWIDTH setting, remove the LRECL from the SYSPRINT DD in job steps that execute DSNTEP2 or DSNTEP4.

Format and content of messages

The format and content of run-time messages is different for PL/I applications that run with Language Environment. Differences include the following items:

- The message number in the message prefix is now four digits instead of three digits.
- The message severity in the message prefix can now be C, E, I, S, or W.
- The message text of some mixed-case English and Japanese messages has been enhanced.

You must modify your applications if they analyze the run-time output. See *z/OS Language Environment Programming Reference* for more information about using and handling messages.

VisualAge PL/I for OS/390 object compatibility

Certain restrictions apply to load modules containing a mixture of VisualAge PL/I for OS/390 objects, and objects produced by earlier compilers (for example OS PL/I and PL/I for MVS & VM). For best results, do not mix compiler levels in a load module. See *VisualAge PL/I for OS/390 Compiler and Run-Time Migration Guide* or *Enterprise PL/I for z/OS, V3R9, Migration Guide* for more information.

General considerations

This topic describes other items you should consider when migrating a pre-Language Environment HLL application to an application that conforms to Language Environment.

Return and reason codes

Some return and reason codes will differ when running under Language Environment. JCL and EXECs that are affected by them must be changed accordingly. See *z/OS Language Environment Debugging Guide* information for details about return and reason codes.

Storage reports

The output of the run-time storage report is different when running with Language Environment. For information about the RPTSTG run-time option, see *z/OS Language Environment Programming Reference*. For an example of the storage report, see *z/OS Language Environment Debugging Guide*.

Stream I/O

If you choose the LINESIZE option, and you provide a record size value that is too small to hold the LINESIZE (taking into account the record format and appropriate control byte overhead), the UNDEFINEDFILE condition is raised.

This behavior is different from previous Language Environment releases. In previous releases, the following was true:

For DD SYSOUT= files except SYSPRINT, if you chose the LINESIZE option, and you provided a record size value that is too small to hold the LINESIZE (taking into account the record format and appropriate control byte overhead), the LINESIZE is used to determine a new record size that matched the given LINESIZE. For DD DSN= files, the UNDEFINEDFILE condition is raised.

Appendix. Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you may view the information through the z/OS Internet Library website or the z/OS Information Center. If you continue to experience problems, send an email to mhvrcfs@us.ibm.com or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at:

`http://www.ibm.com/systems/z/os/zos/bkserv/`

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

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

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Programming Interface information

This book documents intended Programming Interfaces that allow the customer to write programs to obtain the services of Language Environment in z/OS.

Trademarks

Bibliography

This section lists the books in the Language Environment library and other publications that may be helpful when using Language Environment.

Language Products Publications

z/OS Language Environment
- *z/OS Language Environment Concepts Guide*, SA22-7567
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Programming Reference*, SA22-7562
- *z/OS Language Environment Customization*, SA22-7564
- *z/OS Language Environment Debugging Guide*, GA22-7560
- *z/OS Language Environment Run-Time Application Migration Guide*, GA22-7565
- *z/OS Language Environment Writing Interlanguage Communication Applications*, SA22-7563
- *z/OS Language Environment Run-Time Messages*, SA22-7566
- *z/OS Language Environment Vendor Interfaces*, SA22-7568
- *z/OS Language Environment Programming Guide for 64-bit Virtual Addressing Mode*, SA22-7569

z/OS XL C/C++
- *z/OS XL C/C++ Language Reference*, SC09-4815
- *z/OS XL C/C++ Compiler and Run-Time Migration Guide for the Application Programmer*, GC09-4913
- *z/OS XL C/C++ Programming Guide*, SC09-4765
- *z/OS XL C/C++ User's Guide*, SC09-4767
- *z/OS XL C/C++ Run-Time Library Reference*, SA22-7821
- *z/OS XL C/C++ Messages*, GC09-4819
- *Standard C++ Library Reference*, SC09-4949

z/OS Metal C Runtime Library
- *z/OS Metal C Programming Guide and Reference*, SA23-2225

Enterprise COBOL for z/OS, V4R2
- *Enterprise COBOL for z/OS, V4R2, Licensed Program Specifications*, GI11-7871
- *Enterprise COBOL for z/OS, V4R2, Customization*, SC23-8526
- *Enterprise COBOL for z/OS, V4R2, Language Reference*, SC23-8528
- *Enterprise COBOL for z/OS, V4R2, Programming Guide*, SC23-8529
- *Enterprise COBOL for z/OS, V4R2, Compiler and Runtime Migration Guide*, GC23-8527

Enterprise COBOL for z/OS, V3R4
- *Enterprise COBOL for z/OS, V3R4, Licensed Program Specifications*, GC27-1411
- *Enterprise COBOL for z/OS, V3R4, Customization*, GC27-1410
- *Enterprise COBOL for z/OS, V3R4, Language Reference*, SC27-1408
- *Enterprise COBOL for z/OS, V3R4, Programming Guide*, SC27-1412
- *Enterprise COBOL for z/OS, V3R4, Compiler and Runtime Migration Guide*, GC27-1409

COBOL for OS/390 & VM
- *COBOL for OS/390 & VM Licensed Program Specifications*, GC26-9044
- *COBOL for OS/390 & VM Customization under OS/390*, GC26-9045

- *COBOL for OS/390 & VM Language Reference*, SC26-9046
- *COBOL for OS/390 & VM Programming Guide*, SC26-9049

Debug Tool
- Debug Tool documentation is available at: http://www.ibm.com/software/awdtools/debugtool/

VS FORTRAN Version 2
- *Language Environment for MVS & VM Fortran Run-Time Migration Guide*, SC26-8499
- *Language and Library Reference*, SC26-4221
- *Programming Guide for CMS and MVS*, SC26-4222

Enterprise PL/I for z/OS
- *Enterprise PL/I for z/OS, V3R9, Licensed Program Specifications*, GC27-1455
- *Enterprise PL/I for z/OS, V3R9, Programming Guide*, SC27-1457
- *Enterprise PL/I for z/OS, V3R9, Language Reference*, SC27-1460
- *Enterprise PL/I for z/OS, V3R9, Migration Guide*, GC27-1458
- *Enterprise PL/I for z/OS, V3R9, Messages and Codes*, SC27-1461

PL/I for MVS & VM
- *PL/I for MVS & VM Licensed Program Specifications*, GC26-3116
- *PL/I for MVS & VM Programming Guide*, SC26-3113
- *PL/I for MVS & VM Language Reference*, SC26-3114
- *PL/I for MVS & VM Reference Summary*, SX26-3821
- *PL/I for MVS & VM Compiler and Run-Time Migration Guide*, SC26-3118
- *PL/I for MVS & VM Installation and Customization under MVS*, SC26-3119
- *PL/I for MVS & VM Compile-Time Messages and Codes*, SC26-3229
- *PL/I for MVS & VM Diagnosis Guide*, SC26-3149

High Level Assembler for MVS & VM & VSE
- *Programmer's Guide, MVS & VM Edition*, SC26-4941

Related Publications

CICS
- *CICS Transaction Server for z/OS Installation Guide*, GC34-6224
- *CICS Operations and Utilities Guide*, SC34-6229
- *CICS Problem Determination Guide*, SC34-6239
- *CICS Resource Definition Guide*, SC34-6228
- *CICS Application Programming Guide*, SC34-6231
- *CICS Application Programming Reference*, SC34-6232
- *CICS System Definition Guide*, SC34-6226

DB2
- *Database 2 Application Programming and SQL Guide*, SC26-4377

DFSMS/MVS
- *z/OS MVS Program Management: User's Guide and Reference*, SA22-7643
- *z/OS DFSMS DFM Guide and Reference*, SC26-7395
-

IPCS
- *z/OS MVS IPCS User's Guide*, SA22-7596
- *z/OS MVS IPCS Commands*, SA22-7594
- *z/OS MVS IPCS Customization*, SA22-7595

DFSORT

- *z/OS DFSORT Application Programming Guide*, SC26-7523

IMS/ESA

- *IMS Version 8: Application Programming: Design Guide*, SC27-1287
- *IMS Version 8: Application Programming: Database Manager*, SC27-1286
- *IMS Version 8: Application Programming: Transaction Manager*, SC27-1289
- *IMS Version 8: Application Programming: EXEC DLI Commands for CICS and IMS Version 8*, SC27-1288

z/OS

- *z/OS Introduction and Release Guide*, GA22-7502
- *z/OS Program Directory*, GI10-0670
- *z/OS Planning for Installation*, GA22-7504
- *z/OS Information Roadmap*, SA22-7500
- *z/OS Hot Topics Newsletter*, GA22-7501
- *z/OS Licensed Program Specifications*, GA22-7503

z/OS ISPF

- *z/OS ISPF Dialog Tag Language Guide and Reference*, SC34-4824
- *z/OS ISPF Planning and Customizing*, GC34-4814
- *z/OS ISPF Dialog Developer's Guide and Reference*, SC34-4821

z/OS UNIX System Services

- *z/OS UNIX System Services User's Guide*, SA22-7801
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803
- *z/OS UNIX System Services Planning*, GA22-7800

z/OS TSO/E

- *z/OS TSO/E Customization*, SA22-7783
- *z/OS TSO/E Programming Services*, SA22-7789
- *z/OS TSO/E System Programming Command Reference*, SA22-7793

Index

Product Number: 5694-A01

Printed in USA

GA22-7565-12