Integrate your PL/I applications with Web services, XML and Java™

Enterprise PL/I for z/OS, Version 3.9

To make your business as agile and responsive as possible, you need to be able to connect your business components end to end with your suppliers, partners, employees, and customers, and you need to position your organization to quickly take advantage of opportunities by responding to challenges in real time.

Unfortunately, many IT systems weren’t designed to address these objectives or to support Web services and service-oriented architecture (SOA) that are essential for transforming an enterprise into a flexible business with an open, integrated operating environment. You could rewrite your applications in a different programming language in order to address these objectives, but rewriting your applications would be expensive and risky, and it could potentially create downtime that you just can’t afford. To remain competitive, you need a complete business strategy to help you modernize, integrate, and manage existing applications, data, and skill sets to ease your organization’s transformation into a more flexible business.

Highlights

- Enables the creation, maintenance, and modernization of business-critical PL/I applications on z/OS systems
- Exploits IBM® z10 architecture for performance improvements
- Improves MACRO, CICS® and SQL preprocessing
- Leverages productivity with new options, attributes, and built-in functions
- Boosts serviceability with new messages
Integrates, modernizes and manages assets with Web services capabilities

With Enterprise PL/I for z/OS® you can leverage more than 30 years of IBM experience in application development to facilitate your new On Demand Business endeavors, helping integrate PL/I and Web-based business processes in Web services, XML, Java, and PL/I applications. This compiler’s interoperability lets you capitalize on existing IT investment while smoothly incorporating new, Web-based applications as part of your organizations infrastructure.

Enterprise PL/I for z/OS is an integral part of the comprehensive application development environment delivered with IBM Rational® Developer for IBM System z® software—providing a robust, integrated development environment (IDE) for PL/I and connecting Web services; Java Platform, Enterprise Edition (Java EE) applications; and traditional business processes.

Facilitates Web interoperability using XML parsing and generation

Enterprise PL/I for z/OS allows existing PL/I transactions to process inbound and outbound XML data directly within the applications. It provides a high-speed parser that enables PL/I programs to parse XML documents in Extended Binary Coded Decimal Interchange Code (EBCDIC), American Standard Code for Information Interchange (ASCII) or Unicode Transformation Format (UTF)-16. Using the IBM PL/I Simple API for XML (SAX) parser, this XML can then be passed to other applications, even those running on other platforms—including IBM IMS™ and IBM CICS environments.

Enterprise PL/I for z/OS also supports the generation of XML using a built-in function, so you’re able to dump the contents of a structure as XML into a buffer. You can use this XML code to enhance existing high-performance IMS and CICS transactions that have been written in PL/I. By enabling these transactions to send and receive XML documents, you’re better positioned to support a business-to-business (B2B) environment.

Exploits IBM z10 architecture for performance improvements

Additional exploitation of the hardware is implemented in the compiler in order to improve performance of the generated code:

- Under ARCH(7) and higher options, CU12, CU14, CU21, CU24, CU41 and CU42 instructions provide faster Unicode conversions between UTF-8, UTF-16 and UTF-32. For example, CU12 converts a UTF-8 string to a UTF-16 string.
- Under ARCH(7) and higher options, TRRT, TROT, TRTO, and TROO instructions provide fast, inline translation between one byte and two byte buffers. For example, TRRT may be used to “uppercase” a UTF-16 string in the absence of surrogate pairs.
- Assignments of like arrays of scalars are now handled as storage copy operations.
- All assignments of BIT VARYING to BIT VARYING and byte-aligned BIT NONVARYING to BIT VARYING are now inlined.
- ROUND and ROUNDDEC built-in functions are now inlined when the argument to round is a Decimal Floating Point (DFP) instead of using a slower library call.
- The detection of the dereferencing of null pointers exploits the new compare-and-trap instruction under ARCH(8).
- The PL/I compiler is now built with ARCH(6) to improve compiler performance.
- BIN(31,31) compares are now improved.
- UVALID is now inlined for strings of length 256 or less.

Improves CICS preprocessing

In Enterprise PL/I for z/OS, with the integrated CICS preprocessor, it is not necessary to run a separate job step that precompiles EXEC CICS statements into PL/I code. Instead, the compile step will handle EXEC CICS statements in the same way that it handles any use of the MACRO facility. Also, since debugging is against the
source code fed to the compiler, you can debug against the source you wrote (rather than what the CICS precompiler produced).

In Enterprise PL/I for z/OS, Version 3.9, the CICS preprocessor supports block-scoping. This adds local CICS declarations to all non-nested procedures.

**Improves SQL preprocessing**

In Enterprise PL/I for z/OS, with the integrated SQL preprocessor, it is not necessary to run a separate job step that precompiles EXEC SQL statements into PL/I code. Instead, the compile step will handle EXEC SQL statements in the same way that it handles any use of the MACRO facility. Also, since debugging is against the source code fed to the compiler, you can debug against the source you wrote (rather than what the SQL preprocessor produced).

In Enterprise PL/I for z/OS, Version 3.9, the SQL preprocessor supports the PL/I rules for the scope of declarations when resolving host variable references through the new SCOPE option. NOSCOPE is the default for compatibility with previous releases.

**Improves MACRO preprocessing**

In Enterprise PL/I for z/OS, Version 3.9 the MACRO preprocessor:

- Leaves %include, %xinclude, %inscan, and %xinscan statements in the compiler listing as comments making it easier for you to locate that code in the listings.
- Provides an additional MACRO construct, %DO SKIP, which makes it possible to support meta-comments.
- Supports the NAMEPREFIX option to allow you to force macro procedures and variables to start with a specified character. This options allows you to enforce naming conventions for macro procedures and variables.

**Leverages productivity with new options, built-ins and attributes**

With the 3.9 release, Enterprise PL/I for z/OS provides many new and improved options to help detect errors:

- The IGNORE option suppresses PUT FILE statements, DISPLAY statements or both statements. You can now use these statements for debug purposes, while easily compiling them out of the production version.
- The NULLSTRPTR suboption of the DEFAULT compiler provides control over whether sysnull or null is assigned to a pointer when the source in assignment is a null string.
- The new MAXGEN option specifies the maximum number of intermediate language statements that should be generated for any one user statement and will cause the compiler to flag any statement where this maximum is exceeded.
- The new ONSNAP option allows you to request the compiler to insert an ON STRINGRANGE SNAP statement or an ON STRINGSIZE SNAP statement into the prologue of a MAIN or FROMALIEN procedure so that a calling chain trace will be generated if the condition occurs.
- The new SHORT suboption of the INITAUTO option limits the INITAUTO option so that it does not duplicate all of the runtime STORAGE option, but does initialize variables that might be optimized to registers.
- The new RTCHECK option generates code to test for the dereferencing of null pointers.
- The RULES option now provides more control over code and identifies where code can be improved:
  - NOPROCENDONLY flags END statements for PROCs that do not name the PROC they are closing
  - NOSTOP flags the use of STOP and EXIT
  - NOLAXQUAL(STRICT) flags variables not qualified with their level-1 name
  - NOLAXSCALE flags declares of FIXED DEC(p,q) and FIXED BIN(p,q) where q < 0 or q > p
  - NOGOTO(LOOSE) allows GOTO statements only if they are in the same block

In Enterprise PL/I for z/OS, Version 3.9, the CICS preprocessor supports block-scoping. This adds local CICS declarations to all non-nested procedures.

Improves SQL preprocessing

In Enterprise PL/I for z/OS, with the integrated SQL preprocessor, it is not necessary to run a separate job step that precompiles EXEC SQL statements into PL/I code. Instead, the compile step will handle EXEC SQL statements in the same way that it handles any use of the MACRO facility. Also, since debugging is against the source code fed to the compiler, you can debug against the source you wrote (rather than what the SQL preprocessor produced).

Improves MACRO preprocessing

In Enterprise PL/I for z/OS, Version 3.9 the MACRO preprocessor:

- Leaves %include, %xinclude, %inscan, and %xinscan statements in the compiler listing as comments making it easier for you to locate that code in the listings.
- Provides an additional MACRO construct, %DO SKIP, which makes it possible to support meta-comments.
- Supports the NAMEPREFIX option to allow you to force macro procedures and variables to start with a specified character. This options allows you to enforce naming conventions for macro procedures and variables.

Leverages productivity with new options, built-ins and attributes

With the 3.9 release, Enterprise PL/I for z/OS provides many new and improved options to help detect errors:

- The IGNORE option suppresses PUT FILE statements, DISPLAY statements or both statements. You can now use these statements for debug purposes, while easily compiling them out of the production version.
- The NULLSTRPTR suboption of the DEFAULT compiler provides control over whether sysnull or null is assigned to a pointer when the source in assignment is a null string.
- The new MAXGEN option specifies the maximum number of intermediate language statements that should be generated for any one user statement and will cause the compiler to flag any statement where this maximum is exceeded.
- The new ONSNAP option allows you to request the compiler to insert an ON STRINGRANGE SNAP statement or an ON STRINGSIZE SNAP statement into the prologue of a MAIN or FROMALIEN procedure so that a calling chain trace will be generated if the condition occurs.
- The new SHORT suboption of the INITAUTO option limits the INITAUTO option so that it does not duplicate all of the runtime STORAGE option, but does initialize variables that might be optimized to registers.
- The new RTCHECK option generates code to test for the dereferencing of null pointers.
- The RULES option now provides more control over code and identifies where code can be improved:
  - NOPROCENDONLY flags END statements for PROCs that do not name the PROC they are closing
  - NOSTOP flags the use of STOP and EXIT
  - NOLAXQUAL(STRICT) flags variables not qualified with their level-1 name
  - NOLAXSCALE flags declares of FIXED DEC(p,q) and FIXED BIN(p,q) where q < 0 or q > p
  - NOGOTO(LOOSE) allows GOTO statements only if they are in the same block
In Enterprise PL/I for z/OS, Version 3.9, the compiler no longer flags seemingly unneeded %INCLUDE but it now flags:

- Code where the result of a FIXED operation has a scale factor less than zero.
- ENTRYs used as functions but declared without the RETURNS attribute.
- The use of a duplicate ORDINAL in a SELECT statement. The new and unique message IMB2623I is issued.
- Parameters declared inappropriately as BYVALUE, for example, declaring a FIXED DEC parameter BYVALUE.
- FIXED DECIMAL add and multiply operations that might raise the FIXEDOVERFLOW condition.

To simplify compiler options:

- The COMPACT option is dropped
- The default setting for DEFAULT(REORDER/ORDER) is changed to DEFAULT(REORDER) to simplify the options that you select to improve performance
- The TUNE option is dropped

New built-in functions also save you time:

- MEMCU12, MEMCU14, MEMCU21, MEMCU24, MEMCU41 and MEMCU42 built-in functions provide faster Unicode conversions between UTF-8, UTF-16 and UTF-32
- PLITRAN11, PLITRAN12, PLITRAN21, and PLITRAN22 built-in functions provide fast, inline translations between UTF-8 and UTF-16
- The USURROGATE built-in function tests if a CHAR or WCHAR string contains any UTF surrogate pairs
- You can include DFP in restricted expressions when using math built-in functions
- The ROUNDDEC built-in function takes a DFP number and rounds it at the nth decimal digit, rather than at the nth digit as provided by the ROUND built-in function

As well with the 3.9 release, Enterprise PL/I for z/OS provides support for INONLY, INOUT, and OUTONLY attributes so that they are enabled to improve the documentation of function prototypes and the flagging of uninitialized variables.

The following date patterns with blank suppression are also new with the 3.9 release of Enterprise PL/I for z/OS:

- ZY-ZM-ZD
- ZM/ZD/ZY
- ZD.ZM.ZY
- YY-ZM-ZD
- ZM/ZD/YY
- ZD.ZM.YY

**Boost serviceability with new messages**

The Enterprise PL/I for z/OS, Version 3.9 compiler provides new serviceability messages:

- When the compiler cannot open a file, the compiler will now, if possible, also include the related C runtime message in the message in the listing.
- If user code requires a DFP conversion at compile time but the compile is running on a machine without DFP hardware, this error will be trapped and a meaningful error issued.
- If the SQL preprocessor is invoked more than once without INONLY as its suboption, then the DBRM library created by the compiler will be empty, and now an E-level message will be issued to warn you about this problem.

**Provides compatibility for PL/I programs and Java components**

Because it supports the Institute of Electrical and Electronics Engineers (IEEE) decimal floating point standard, the Enterprise PL/I for z/OS compiler can receive, manipulate and send Java data without any translation.

Built-in functions provide support for UTF-8 and UTF-16. One example is the ULENGTH function, which returns the number of UTF-8 or UTF-16 characters in a CHAR or WIDECHAR string, respectively. A second important example is the USUBSTR function which returns the UTF-sensitive substring of a CHAR or WIDECHAR string.
To further improve Java interoperability, Enterprise PL/I for z/OS provides a thread-safe PL/I library and multithreading statements (ATTACH, WAIT, DETACH) as part of the PL/I language supported by the compiler.

**Ease into migration**

Enterprise PL/I for z/OS gives you a migration path from OS PL/I V2 and PL/I for MVS and VM compilers. Our Compiler and Runtime Migration Guide provides you with all the information that you might need to move your applications to a new run-time (run-time migration) and to compile your source programs with the new compiler (compiler migration). Migrating to the new compiler allows your existing applications to take advantage of new functions.

**Workstation-based development**

Rational Developer for System z provides an interactive, workstation-based environment to help you create, maintain, and reuse applications. Rational Developer for System z includes support for traditional development using PL/I, but also has the ability to generate Web services interfaces from PL/I constructs to ease creation of Web services from existing PL/I applications.

Rational Developer for System z provides a workstation interface to Debug Tool, and is also integrated with IBM File Manager and Fault Analyzer. File Manager integration enables you to access Keyed Sequence Data Set (KSDS) files from the Rational Developer for System z workbench, and gives you the ability to browse and update data sets. By integrating with Fault Analyzer, Rational Developer for System z enables you to browse Fault Analyzer ABEND reports on CICS, IMS, batch, Java, WebSphere®, and other run times.

**PL/I across platforms**

Enterprise PL/I for z/OS is part of a family of compatible compilers, application development tools, and maintenance tools. Along with Enterprise PL/I for z/OS, IBM offers PL/I compilers for multiple platforms as well as IBM File Manager, IBM Fault Analyzer, and Debug Tool. As mentioned previously, the recommended workstation-based development environment is Rational Developer for System z.

![Rational Developer for System z workbench](Figure 1: Rational Developer for System z workbench)
For more information
To learn more about how IBM Enterprise PL/I for z/OS, Version 3.9 can help your enterprise in the transformation to a flexible, performance-driven business, contact your IBM representative or IBM Business Partner, or visit: ibm.com/software/awdtools/pli/plizos

To learn more about IBM Rational Developer for System z software, visit: ibm.com/software/awdtools/rdz

For more information visit www.ibm.com/software/awdtools/pli/plizos