

Welcome to this IBM podcast, Leverage IBM COBOL, PL/I, and C/C++ compilers to optimize Investments on System z.

Compilers are a bridge between user application and the systems that runs business. It helps improve programmer productivity and optimize application performance.

IBM compilers on z/OS are designed to unleash the full power of IBM System z architecture and middleware. They target application development and deployment on z/OS. They support batch processing, legacy file systems, and provide a major programming interface for CICS, IMS and DB2.

IBM is very experienced in delivering leading edge compilers on System z. We have been delivering COBOL compiler for over 40 years, PL/I compiler for over 30 and C and C++ compilers for over 20 years. These compilers grew with IBM mainframe hardware and middleware.

Compilers are very critical to the success of System z platform. COBOL and PL/I currently run the world's most business critical applications. C/C++ is a dominant implementation language of portable applications and middleware and its usage is increasing on z/OS. All compiler products support the latest System z architecture.

IBM extended the COBOL and PL/I programming languages to support application modernization. This enables customers to leverage modern technology in legacy applications. Legacy applications are crucial to business. A considerable amount of money and resources have been invested in the development and enhancements over a long period of time. Applying modern technologies without the requirement to re-write or re-host these applications reduce risk, cost and time to market. This can directly translate to improved bottom-line success for you.

Enterprise COBOL for z/OS enables the integration of existing applications with web applications. It supports Java interoperability with object-oriented COBOL syntax. It also provides you access to enterprise beans that run on WebSphere Application Server or J2EE-compliant EJB server. It supports processing of XML documents by providing built-in language support for high speed XML parsing and generation. COBOL XML parsing can be offloaded to zAAP specialty processors on System z10. Enterprise COBOL also supports the latest middleware and tools like DB2, CICS, and Debug tools.

The latest version of Enterprise COBOL for z/OS (v4.2) supports validation of XML document parsing against a schema while it processed by the COBOL application. It also provides significant performance improvement when using z/OS XML System Services parser. It supports the latest DB2 9, and CICS 4.1. This new COBOL product is also enhanced to support Java 5 and Java 6 runtime, enhancing capability to integrate COBOL applications with Web applications.

Enterprise PL/I for z/OS support integration of PL/I applications and web-based business processes in Web services, XML documents, and Java™ applications. It provides PL/I built-in subroutines for high speed XML parsing and generation. XML parsing supports offloading to zAAP specialty processors on System z10. This provides significant cost savings. Enterprise PL/I supports the latest middleware and tools like DB2, CICS, and Debug tools.

Enterprise PL/I for z/OS fully exploit the new System z10 architecture, including the Decimal Floating Point Unit. The latest version, v3.9 improved support for internationalization and I/O capabilities. It also improved support for middleware, DB2 and CICS, and delivered a number of user requirements. For example, improved performance in array and bit assignment, provide more rules to enforce coding standards, and new compiler messages to help tune application performance...

IBM z/OS XL C/C++ compiler is a separately charged feature of the z/OS operating system. It supports both 31-bit and 64-bit applications and provides full exploitation of the new System z10 processor, including the Decimal Floating Point Unit. z/OS XL C/C++ compiler provides a feature to support system programming capabilities. This new “Metal C” option allows you to use C/C++ in place of assembler language for system program development. It is capable of generating optimized assembler code which can take experienced mainframe assembler programmers a relatively long time to develop. “Metal C” code is highly portable amongst System z platforms. To move from one platform to another, all you need is to recompile “Metal C” source to target the new platform. This requires no coding effort and therefore significantly reduces cost, risk and time to market.

The latest version of z/OS XL C/C++, v1.11, delivers significant performance improvements. It improved the performance of overall SPECint2006 benchmark by almost 9%. It also delivered a number of usability enhancements. For example, improved readability of compiler listings, support debug without source code, and generation of pseudo-C listing to assist debugging of optimized code and performance tuning.

The z/OS Language Environment provides a common foundation to run programs written using different programming languages. Programs developed with Enterprise COBOL for z/OS, Enterprise PL/I for z/OS, z/OS XL C/C++, and Assembler can perform inter-language calls with one another. This enables you to fully utilize the strengths of different programming languages within the same application.

In summary, IBM compilers are designed to exploit System z hardware and middleware (i.e. CICS, DB2, IMS...). Rational is committed to delivering leading-edge compilation technology to maximize application performance, improve programmer productivity, and shorten development cycle which could result in lower Total cost of ownership and higher return on IT investment.

For more information on Enterprise COBOL, Enterprise PL/I and z/OS XL C/C++, Please visit the corresponding product web pages:

- COBOL <http://www.ibm.com/software/awdtools/cobol>
- PL/I <http://www.ibm.com/software/awdtools/pli>
- C/C++ <http://www.ibm.com/software/awdtools/ccompilers>

Please also visit our community site, Rational Cafes, at [www.ibm.com/rational/cafe](http://www.ibm.com/rational/cafe).

Thank you very much for listening.