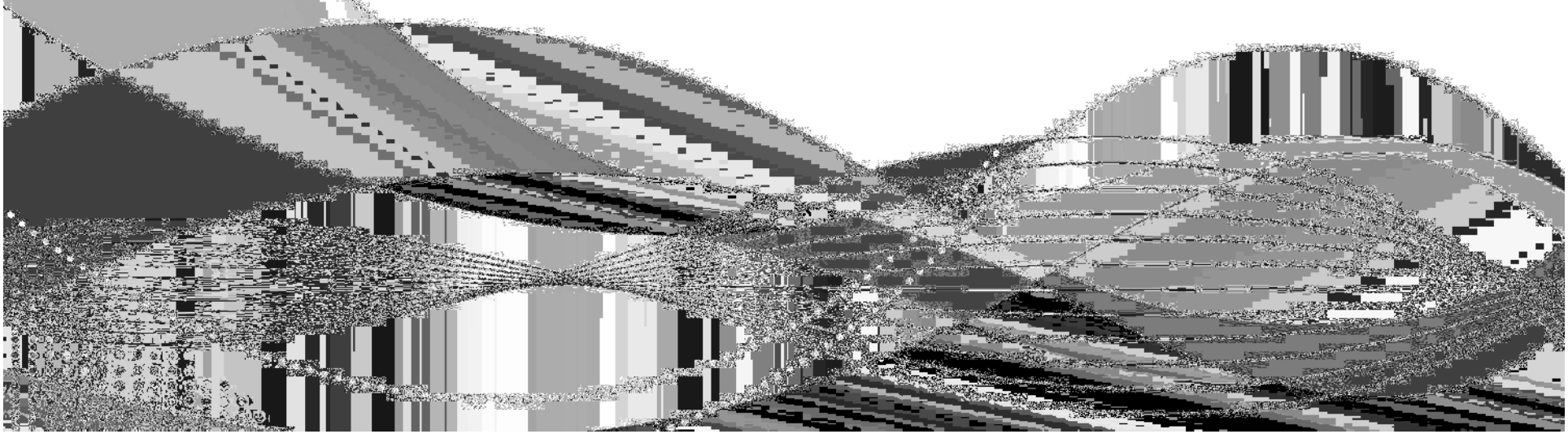
# Full Speed Ahead with COBOL Into the Future

**Speaker Name: Tom Ross**
**Speaker Company: IBM**

**Date of Presentation: August 8,2011**
**Session Number: 9652**



# Agenda

- COBOL and Java and … DB2!
- COBOL development team in IBM
- What are we up to these days?

# COBOL and Java and … DB2

# COBOL and Java and … DB2

- COBOL shipped COBOL and Java interoperability in 2001
- As users started using these features they tried to use these applications in more complex environments (IE: More components/subsystems such as DB2)
- Java has a different model for connecting to DB2 then 'traditional' z/OS programs
  - JDBC rather than CALL attach
  - type-2 z/OS connectivity or type-4 z/OS connectivity
- When COBOL and Java were mixed, they ran as a single application, but got 2 separate connections to DB2
- Either COBOL or Java could fail without the other part of the application failing

# COBOL and Java and … DB2

- What are users finding when using the type-2 z/OS connectivity with mixed COBOL, Java and SQL?
  - We are using Enterprise COBOL from a batch job and trying to call a Java program that uses DB2. Since the COBOL program is already doing some DB2 work it already has a RRS connection. When the Java program is invoked it tries to create a connection and fails.
  - I am using the COBOL OO features to invoke a Java object method. The COBOL program is using RRSAF attach and issuing EXEC SQL commands successfully prior to invoking the Java object. The Java object is also using DB2. However, when the Java object attempts to connect to DB2 using the Universal type 2 connector (jcc) it is receiving an InvalidNumberFormatException.

# COBOL and Java and … DB2

- What are users finding when using the type-4 z/OS connectivity with mixed COBOL, Java and SQL?
  - Java connection to DB2 can be established, but the connection is independent of any RRSAF attachments that may be in use for COBOL module support. This means that the JDBC transaction will be based on a remote DRDA thread and not have shared transaction scope. Huh?
  - More simply, if the COBOL fails and the Java succeeds, then part of the application will rollback updates (COBOL) and part will commit the database updates (Java).
  - In addition, there is no way for COBOL condition handling to know about Java exceptions, and vice versa
  - What is a COBOL-Java-DB2 interop user to do?

# z/OS Batch Runtime Environment

- An integrated part of z/OS R13
- A new option for running batch work
- Provides a managed environment for integration of Java and COBOL with RRS compliant resource managers
- Consistent with IBM WebSphere based batch
  - A subset of the WebSphere programming model
  - Incorporated in the OS
- DB2 is the only supported resource manager currently

# Batch Execution Runtime Environment Java COBOL with DB2 Interoperability

- Ability to extend COBOL/DB2 applications with Java (or Java/DB2 applications with COBOL):
  - Requires local attach z/OS DB2 connection sharing for common DB2 access
  - Requires Transactional integrity among the application components
- A generalized solution in a pure batch environment, without requiring a specific run time or middleware. It does not require an application server like CICS or DB2 Stored Procedures.
- Requires little or no changes to existing code!
  - Only requires special callbacks for commit/rollback
- Enables interoperability between COBOL, Java and DB2 in batch

# Batch Execution Runtime Environment Java COBOL with DB2 Interoperability

- Special callbacks for commit/rollback
  - Use instead of EXEC SQL COMMIT or ROLLBACK
- z/OS Batch Runtime provides helper functions to simplify the COMMIT/ROLLBACK processing
  - These helper functions are shipped as Java
  - For Java, methods for commit and rollback functions are available with the following package: com.ibm.batch.spi.UserControlledTransactionHelper
- For commit: UserControlledTransactionHelper.commit()
- For rollback: UserControlledTransactionHelper.rollback()

# Batch Execution Runtime Environment Java COBOL with DB2 Interoperability

- Certain restrictions apply to COBOL and Java applications in the z/OS Batch Runtime environment.
  - COBOL applications must not use the STOP RUN statement. Using STOP RUN prevents the z/OS Batch Runtime environment from receiving control. Instead, use the GOBACK statement to end the COBOL application.
  - A COBOL main program will no longer be the first program entered. COBOL specific runtime options might be affected.
  - Java applications must be single threaded and must not use the system.exit method. Using the system.exit method ends the JVM and prevents the z/OS Batch Runtime environment from receiving control. Instead, end the main Java procedure with a simple return statement.

# COBOL development team in IBM

# COBOL development team in IBM

- Compiler development now consolidated into the Rational division of IBM
- After years of shrinking, COBOL development expanded!
- IBM has doubled the size of COBOL development team
- Renewed investment, interest and commitment
- COBOL is the language of the Future!

# What is IBM COBOL development up to these days?

# Please Note:

- **IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.**

  **The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.**

# What is IBM COBOL up to?

- Sometimes people take 'no news' as 'bad news'
- It may be some time before the next IBM COBOL compiler is shipped for z/OS
- This is actually good news, because IBM COBOL development is heads down working hard on exciting new technology!

# Moving forward…

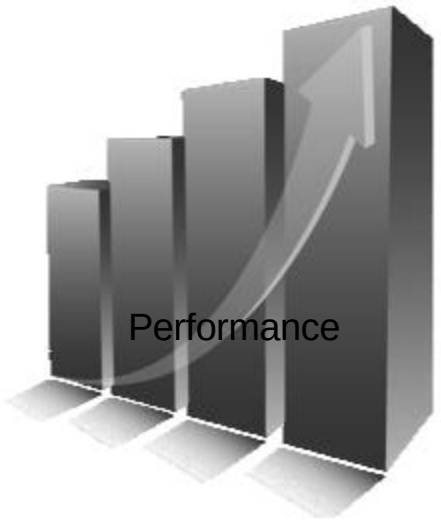
- **COBOL, C/C++ and PL/I are strategic on System z**
  - Run the world's business critical applications
  - IBM has strong commitment to COBOL, C/C++, and PL/I languages
    - Continue to provide additional value to customers
      - *Improve performance*
        - *exploitation of z/Architecture & IBM Middleware*
        - *optimization technology*
      - *Improve portability and programmer productivity*

- **Modernize compiler infrastructure for COBOL**
  - Improve performance and exploitation of System z architecture

**IBM reserves the right to change strategy and plans at any time.**

# Modernizing Enterprise COBOL infrastructure

- **Our intent:**
  - Incorporate leading-edge optimization and code-generation technology to Enterprise COBOL for z/O
    - Improve performance, and maximize machine utilizat
    - Reduce cost of ownership
  - Improve delivery of z/Architecture exploitation
    - Provide solid foundation to support new hardware features (e.g. decimal floating point, 64 bit…), and future System z processors
  - Continue to improve capabilities for modernizing business critical applications and creating new applications
    - XML enhancements
    - Usability and problem determination enhancements




*** IBM reserves the right to change strategy and plans at any time.**

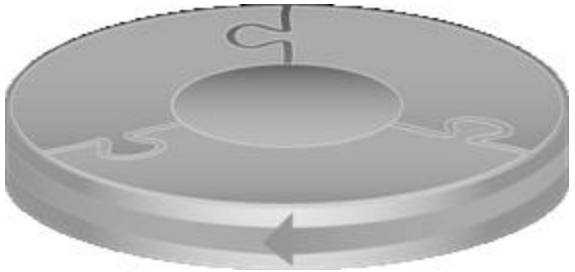
# Compatibility Objectives

We intend to provide source and binary compatibility

- No need to recompile old code
  - Support linking together of “old” and “new” code for correct programs
    - *Old applications will be able to call new applications.*
    - *New applications will be able to call old applications.*

We intend to run “Closed” beta program (Date: TBD)

- Send note to Roland Koo (rkoo@ca.ibm.com) if you are interested in participating



**IBM reserves the right to change strategy and plans at any time.**

# I can't share all the details yet, but…

- Proposed changes to COBOL language
  - Maybe eliminate some obsolete features
    - Anyone using MLE (DATE data types?)
- Hardware exploitation
  - As soon as it comes out!
  - COBOL could have the ARCH compiler option

**IBM reserves the right to change strategy and plans at any time.**

# I can't share all the details yet, but…

- Performance improvements
  - Could use DFP instructions to implement decimal arithmetic
  - Double-word binary arithmetic
  - Relative addressing instructions
    - Good bye BLL cells!
- Debug Tool support improvements
  - Anyone have issues with finding SYSDEBUG files or large load modules footprint?
  - Show me what I wrote, not how COBOL interpreted it
  - Calculating lengths and locations with ODO - fixed!

**IBM reserves the right to change strategy and plans at any time.**