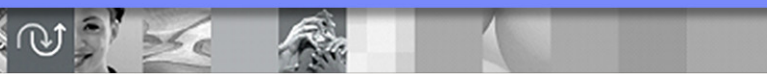





You're
invited



Discovering the Value of IBM Rational Developer for System z version 7.1 (RDz)

AN IBM PROOF OF TECHNOLOGY

INTRODUCTION

[RATIONAL® DEVELOPER FOR SYSTEM Z™ VERSION 7.1](#) speeds efficiency of traditional mainframe, Web development and composite applications.

COBOL, PL/I, C, C++ and Java™ developer communities are more productive when they take advantage of RDz functions. RDz consists of a common workbench and an integrated set of tools that support end-to-end, model-based application development, runtime testing, and rapid deployment of On Demand applications. It offers an integrated development environment (IDE) with advanced, easy-to-use tools and features to help WebSphere®, CICS®, and IMS™ developers rapidly design, code, and deploy complex applications.

RDz also includes wizards that provide service flow modeling capabilities to enable business service integration for CICS applications.

Those tools provide a graphical modeling interface to create business services that may be invoked as Web Services, by composing a sequence of CICS application interactions.

RDz enables enterprise solution architects, integration developers, and CICS application specialists to graphically implement a service-oriented architecture (SOA).

OBJECTIVE

This Proof of Technology session provides customers with basic skills and hands-on exposure to the major features of the Rational Developer for System z version 7.1. Attendees will complete structured walkthrough labs that demonstrate the major features and new functionality of RDz. The labs also provide an introductory hands-on experience to using RDz on building a system z-based Service Oriented Architecture (SOA).

- The main points to be covered are
- How to code, test and debug a simple COBOL or PL/I program that access DB2® without connection to the mainframe, using RDz
- How to code, compile and debug using a remote connection to z/OS®, eliminating the need for TSO/ISPF.
- How to create a Java client from existing z/OS CICS or IMS COBOL programs using J2C connectors, and then test it.
- How to transform and test existing COBOL programs to understand XML enabling service-oriented architecture (SOA) access to CICS Transaction Server or IMS.
- How to create, test and debug z/OS DB2 Stored Procedures using COBOL or PL/I
- How to use the BMS Editor for CICS or MFS Editor for IMS with RDz
- How to generate COBOL/CICS/DB2 program from an existing z/OS Db2 table..
- How to use the Enterprise Service Tools Service Flow modeling capability of RDz to create a business service that may be deployed to CICS and invoked as a Web service...

AUDIENCE

This Proof of Technology is intended for Architects, Technical Specialist or Developers who are evaluating the IBM® Software Development Platform and Service Oriented Architecture offerings for use in their enterprises...

COST

This session is offered free of charge. Complimentary refreshments including continental breakfast and lunch will be provided. However, participants are responsible for their own business travel expenses.

SCHEDULE

For your convenience, registration and continental breakfast will begin at 8:30 AM. The session will start at 9:00 AM and end at approximately 5:00 PM.

CONTACT FOR INFORMATION

To enroll in this Proof of Technology, please contact your IBM Software Sales Representative.



AGENDA

Agenda: This session is designed to give a high level, hands-on view of RDz version 7.1. It covers the most important features of the product. We have 3 basic labs and many exploratory labs. All students must complete the basic labs and based on their interest they will elect which exploratory labs to complete.

Lab 1A/1B – Using RDz to Work with a Local COBOL or PL/I Program

This lab takes you through the steps of using the z/OS Application Development component of RDz to work with a local COBOL or PL/I program. It familiarizes you with the COBOL and PL/I Editor, the IDE and the integrated debugger. You may choose LAB1A for COBOL or LAB1B for PL/I.

Lab 2 – Using RDz to work with a z/OS Remote System

This lab takes you through the steps of using the Remote System Explorer component of RDz to work with remote mainframe systems. It familiarizes you with the z/OS Application Development environment. If the connection to the mainframe is available, you will define a remote z/OS system, set up an MVS™ project, edit, compile, and debug a COBOL application. This Lab uses COBOL as a program example, but all would also apply for PL/I users. C/C++ users also would take advantage since most of the features also would apply for.

Lab 3 - Creating CICS Services from COBOL program using WebSphere Developer for System z

This lab will take you through the steps of using the Enterprise Service Tools (EST) component of WebSphere Developer for System z V7.1 to transform an existing CICS COBOL application so that it can be accessed through the CICS Web Services facilities of CICS Transaction Server for z/OS V3.1.

Lab 4-E - Developing and Testing CICS BMS COBOL Application that access local DB2

This lab will take you through the steps of using RDz to build an application consisting of CICS, COBOL and DB2. The application can be run and debugged using the TXSeries and DB2 UDB. The RDz BMS Map Editor will also be used to show how a map can be created and deployed in support of CICS transactions. This lab will also guide you through the steps of integrating RDz with TXSeries®. While the lab will be done locally using TXSeries, that is, with the development artifacts being hosted on the workstation, the steps will be similar if the artifacts reside on a z/OS system.

Lab 5A-E – Calling COBOL/IMS using J2C connectors (JCA Architecture)

The J2C tools available in RDz can expose the interaction specifications and properties of the IMS resource adapter along with generating a java class that accesses IMS transactions. Once the J2C Java Bean is created, it can easily be consumed by a web page (JSP), web service, or Enterprise Java Bean. This lab exercise will help familiarize you with the J2EE Development environment and the J2C. In this Lab, we will be working with the sample Phonebook IMS transaction (IVTNO), which is one of the IMS installation verification programs shipped with IMS. As an optional exercise, you will also experiment with creating a web service from the generated J2C Java Bean.

Lab 5B-E – Calling LOCAL COBOL/CICS (TXSeries) using J2C connectors (JCA)

In this lab, you will work with a COBOL transaction that returns the customer information for a given customer number. The existing COBOL program has a communication area (commonly referred to as COMMAREA) where we pass in a customer number and get the customer information for that customer (First Name, Last Name, Address, etc.). We will use the TXSeries CICS Transaction Gateways (CTG), but the steps are the same as if COBOL was in the CICS z/OS.

Lab 5C-E – Calling z/OS COBOL/CICS using J2C connectors (JCA Architecture)

In this lab exercise we will run a 'test' WAS on our workstation in our RDz environment. The generated Java code uses the J2C (J2EE Connector Architecture) to communicate to a CICS Transaction Gateway (CICS TG), and then on to CICS. For this lab exercises, the CICS TG will be running on z/OS, but like WAS, the CICS TG can run on many supported platforms. The target program supplies customer information and is a COBOL program under CICS.

Lab 6-E - Using the Enterprise Service Tools Service Flow capabilities of WebSphere Developer for System z to Model a Business Service from Existing CICS Applications

This Lab will take you through the steps of using the Enterprise Service Tools (EST) of RDz to develop service flow projects. This Lab will focus on the bottom-up approach by creating a service from an existing CICS terminal application. The CICS TS 3.1 sample Catalog Manager application will be used. This application is invoked from a CICS terminal through the EGUI transaction. You will be creating a service that will return a list of items that are available from the catalog.

Lab 7-E – Create, Test and Debug COBOL DB2 Stored Procedure

This lab takes you through the steps of generating a COBOL program that will execute in the z/OS as DB2 Stored Procedure. You will create the code, load in the z/OS DB2 and test it.

Lab 8-E – Using the Database Application Generator RDz wizard

The System z Database Application Generator enables you to use an existing DB schema to rapidly create a z/OS data access layer for Web Services. In addition, you can work directly from a UML model.

Lab 9-E – Using IMS/MFS Wizards and Editor with RDz

This Lab shows how to create a new MFS device format definition and the associated message format definitions using MFS wizards and the MFS Editor