Modern z/OS application
development with RDz

GSE NL COBOL working group
Woerden - 14/05/2009
Gie Indesteege - ABIS Training & Consulting
Welcome

ABIS Training & Consulting
www.abis.be

Gie Indesteege
• trainer and consultant
• president of BeNeLux GSE working group EGL/RDZ
  Enterprise Generation Language/
  Rational Developer for System z
Developing applications for the mainframe environment has been done for years, based on the ISPF dialog manager facilities and tools.

This well known application development life cycle is still used, but ... modern application development tools make it possible to optimise, enhance, facilitate all these aspects.

Moreover integration of these aspects will shorten the development process.

RDz (Rational Developer for System z), based on the Eclipse foundation, will help to make application development life easier and perhaps better.

This presentation will give you an overview of the architecture and possibilities of the development tool.
Agenda

• Traditional application development
• Architecture of the Rational Developer for System z
• Project setup
• Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
• Support of development process (synchronisation, remote debugging, build, change management)
• Integration aspects
• Q & A
Traditional application development for z/OS

Application development life cycle
1. analysis and design

Repeat until OK
2. coding of COBOL sources -> ISPF/PDF editor
3. compilation/link-edit -> JCL procedures
4. check compile using SDSF
5. data preparation -> data manipulation tools
6. test/debug
   - runtime environments (batch, CICS, IMS)
   - data stores (VSAM, DB2, DL1, ...)

Approval
7. stage into acceptation/production
Example

Edit COBOL source -> save -> Edit compile JCL -> submit

```
EDIT       TBISTS0.COBUT.SOURCE(HELLO) - 01.04          Columns 00001 00072
Command ==>                        Scroll ==> CSR
000001   01    WELCOME-MSG  PIC X(50) VALUE '----------'
000002    *  
000003   PROCEDURE DIVISION.
000004         *-----------------------*
000005   MAIN-PROGRAM.
000006         *-----------------------*
000007   PERFORM INITIALISATION
000008   PERFORM DISPLAY-WELCOME-MSG
```

```
EDIT       TBISTS0.COBUT.SOURCE(COMPBAT) - 01.04          Member COMPBAT saved
Command ==>  sub_              Scroll ==> CSR
000007 //*************************************************************************
000008 //              SET PGMNAM=HELLO
000009 //*
000010 //*               COMPILE THE COBOL PROGRAM
000011 //*
000012 //COB           EXEC PGM=IGYCRCTL,PARM='APOST,TERM,FLAG (W,W),TEST(STMT,SYM)'
000013 //STEPLIB       DD DSN=SYS1.COBOL.SIGYCOMP,DISP=SHR
```
Example (cont.)

Swap -> Check SDSF - find Job

<table>
<thead>
<tr>
<th>Display</th>
<th>Filter</th>
<th>View</th>
<th>Print</th>
<th>Options</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDSF JOB DATA SET DISPLAY - JOB TB00127P (JOB27433) LINE 1-6 (6)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMMAND INPUT ==&gt;</td>
<td>SCROLL ==&gt;</td>
<td>CSR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PREFIX=TB*</td>
<td>DEST=(ALL)</td>
<td>OWNER=*/</td>
<td>SYSNAME=</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NP</th>
<th>DDNAME</th>
<th>StepName</th>
<th>ProcStep</th>
<th>DSID</th>
<th>Owner</th>
<th>C Dest</th>
<th>Rec-Cnt</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JESMSGL</td>
<td>JES2</td>
<td></td>
<td>2</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>JESJCL</td>
<td>JES2</td>
<td></td>
<td>3</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td></td>
<td>JESYSMS</td>
<td>JES2</td>
<td></td>
<td>4</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYSPRINT</td>
<td>COB</td>
<td></td>
<td>101</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>113</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYSTEMS</td>
<td>COB</td>
<td></td>
<td>102</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYSPRINT</td>
<td>LKED</td>
<td></td>
<td>103</td>
<td>TB00127</td>
<td>X LOCAL</td>
<td>182</td>
<td></td>
</tr>
</tbody>
</table>

find code line in error (remember) -&gt; swap to source

... repeat until OK -&gt; prepare TEST

- data
- runtime environment
Traditional development

**PRO**
- well known development environment
- central management
  - mainframe RAS - security, concurrency control, backup/recovery
- shareable (source, procedures, data, ...)

**CON**
- long cycle
- switching between environments
- rigid procedures, not flexible
Agenda

• Traditional application development
• Architecture of the Rational Developer for System z
• Project setup
• Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
• Support of development process (synchronisation, remote debugging, build, change management)
• Integration aspects
• Q & A
Architecture of Rational Developer for System z

Based on the Eclipse workbench (www.eclipse.org)
- open, portable and universal tooling platform
- manage complexity of different
  - runtime environments
  - operating systems
  - workgroup servers
- provides frameworks and services and tools for building plug-in tools
- all functionality is provided by plug-ins
- role-based development (single repository)
  - web designer, COBOL programmer, administrator, architect
- file-based IDE
- team development
Modern z/OS application development with RDz

Architecture of RDz (cont.)

XML Services for the Enterprise
- SOA access to CICS V3.1 and IMS V6 COBOL applications
- Bottom-up or meet-in-the-middle COBOL to XML mapping support
- Integrated COBOL XML converters, XML schemas, and WSDL generation

Rational Developer for System z

z/OS Application Development
- Connect to z/OS systems
- Work with z/OS resources like COBOL programs, JCL, etc.
- Interact with the Job Entry Subsystem (JES) to submit jobs, monitor jobs, and review job output
- Perform dataset management actions like allocating datasets and migrating datasets
- Perform typical edit, compile, and debug tasks on remote z/OS resources from the workstation

XML Services for the Enterprise

BMS Map Editor
- Visually create and modify BMS Map sets
- Work with local or remote maps

DB2 Stored Procedure for COBOL and PL/I
- Create DB2 stored procedures on z/OS in either COBOL or PL/I
- Build and catalog support for the DB2 stored procedure
- Debug z/OS based stored procedures from workstation

EGL COBOL Generation
- Deploy EGL applications to zSeries CICS or batch environments
- Connectivity to CICS through JCA
- JSF UI components integrated with CICS services

IBM Rational Application Developer
Features of RDz

intended for (legacy) enterprise developers

• development of large enterprise applications by connecting web applications to enterprise business logic

• supports multiple technologies:
  • Java SE and Java EE (JSPs, servlets, Struts, JSF, EJBs)
  • web applications (HTML, CSS, JavaScript, AJAX)
  • XML and web services
  • COBOL, PL/I, Assembler
  • Enterprise Generation Language (EGL) for Java and COBOL

• integration with z/OS and subsystems (JES, CICS, DB2, ...)
  • BMS map editor
  • DB2 stored procedures
  • JES spool access ...
Agenda

• Traditional application development
• Architecture of the Rational Developer for System z
• Project setup
• Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
• Support of development process (synchronisation, remote debugging, build, change management)
• Integration aspects
• Q & A
Project setup in RDz

• perspectives
  - Remote System Explorer -> connection to host
  - z/OS -> development of legacy applications

• views
  - z/OS (MVS) projects
  - source outline
  - z/OS file system mapping
  - remote system info (details, error list, shell)
  - Job monitor
  - TSO commands

• editors
  - LPEX (z/OS, basic or remote systems)
Workstation based development

Remote development
- working on local copy of host files (PDS, SDS)
- local syntax checking
- build implies remote compile and link on host
  generate and customise JCL
- remote debug (run on host)
- local error reporting

Local development
- working on local workstation files
- local syntax checking
- local compile and build
- local debug
Connection between RDz and host

Connection to z/OS host

- **Remote System Explorer (RSE)**
  - z/OS file mapping for COBOL, JCL, PL1, ...
  - code page customisation

access to JES spool -> Job monitor

also used for remote debugging -> Debug tool
Agenda

• Traditional application development
• Architecture of the Rational Developer for System z
• Project setup
• Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
• Support of development process (synchronisation, remote debugging, build, change management)
• Integration aspects
• Q & A
Developing COBOL applications with RDz

Set up connection to host

![New Connection dialog box](image)
Developing COBOL applications with RDz (cont.)

Look-up z/OS resources
Developing COBOL applications with RDz

Define associations of remote files
- JCL
- COBOL
- PL/1
- BMS
- REXX

Open file with LPEX editor
- ISPF like editor
- language sensitive highlighting
- syntax checking
- code assist
- outline view
z/OS project definitions

Combine z/OS resources into z/OS projects
z/OS project

Workstation based development of host resources (COBOL, JCL, ...)

1. create MVS project
   via z/OS projects view

2. search PDS on z/OS host
   -> MVS files -> My Data Sets

3. optional: allocate new PDS
   based on High Level Qualifier and data set name
   -> define data set characteristics

4. add PDS to MVS project

5. Edit COBOL source

6. Build the project, based on (main) program entry point

7. Run (or debug) generated load module
Local project

1. Create local project (Workstation COBOL or PLI)
2. Edit COBOL source
3. Build the project, based on (main) program entry point
4. Run (or debug) generated load module (.exe)
Example of local project
Customise build process

Modify properties of file

- use CICS
  - invokes CICS translator
- user DB2
  - invokes SQL pre-compiler
- COBOL compiler options
  - remote COBOL
  - local COBOL
- use IMS
  - specify IMS libraries
Agenda

- Traditional application development
- Architecture of the Rational Developer for System z
- Project setup
- Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
- Support of development process (synchronisation, remote debugging, build, change management)
- Integration aspects
- Q & A
Supporting the development process

• CARMA - Common Access (Host) Repository Manager
  access to Software Configuration Manager (SCM) - e.g. SCLM
    • CARMA repository view
    • check-in/check-out
    • project synchronisation
    • team working

• Remote debugging
  integration with IBM Debug tool

• Build
  Rational Team Concert for System z
Agenda

• Traditional application development
• Architecture of the Rational Developer for System z
• Project setup
• Developing COBOL applications with RDz (analysis, coding, validation, preparation, testing)
• Support of development process (synchronisation, remote debugging, build, change management)
• Integration aspects
• Q & A
Integration aspects

Development of CICS applications
• BMS map editor
• Service Flow Modeler (for CICS) - tech preview

Development of DB2 stored procedures

XML services - enable COBOL programs as web services
• WSDL generation
• type conversion
  - web service
  - CICS SOAP
  - IMS SOAP
Q & A
Thank you

Gie Indesteege
Trainer and Consultant
gindeesteeg@abis.be

thanks you