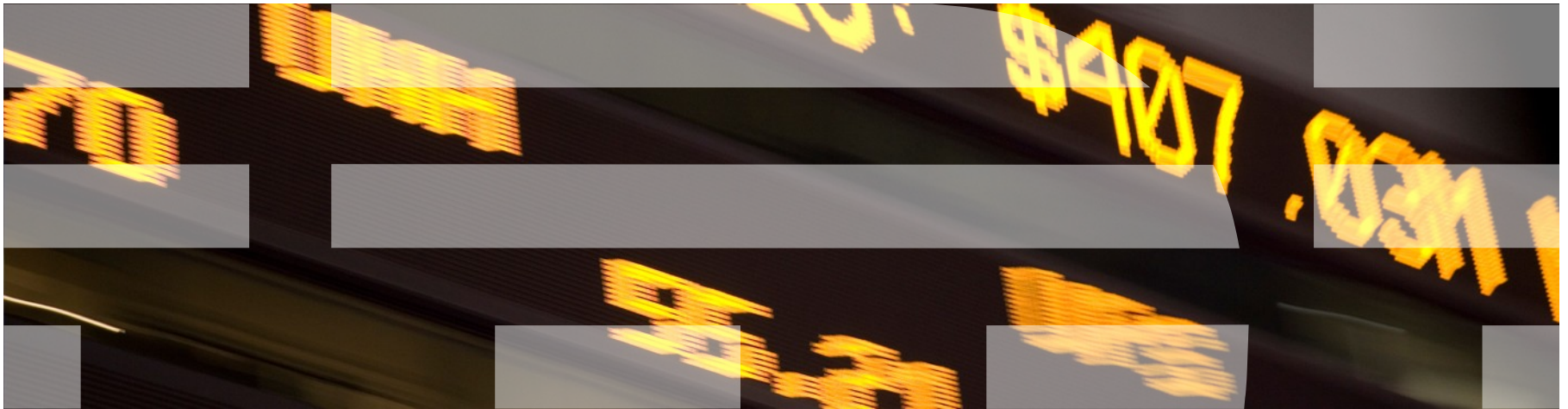


Application Development for z/OS

Not your Father's Green Screen

SHARE Session 8656



Abstract

Ask most people how they write and maintain applications on z/OS and you hear "oh, you use this thing called a green screen" followed by a chuckle.

In reality, application development for zEnterprise applications has been transformed over the past several years to the point where application developers enjoy the same or better features from integrated development environments as programmers who work on other platforms.

Advances in remote system communication and interaction, syntax-highlighting, parsing, and code understanding for Assembler, PL/I, C/C++, and COBOL source code, as well as programming assists such as code snippets and templates are all available to application programmers. Interactive debug of applications, written in multiple programming languages and running in various runtime environments is also possible and can greatly boost programmer productivity.

Come and learn about how these features can enable application developers who are new to the mainframe to interact with, update, and efficiently enhance mainframe applications.

Agenda

- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- Continual Discovery
- Reprise: Application Development is Hard

Agenda

- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- Continual Discovery
- Reprise: Application Development is Hard

Traditional Application Development for z/OS

- study compiler listings (green bar printout) or use ISPF



```

File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT      DDS0001.TEST.COBO(LHOSPEDIT) - 27.26          Columns 00001 00072
***** ***** Top of Data *****
000001      *****
000002      IDENTIFICATION DIVISION.
000003      PROGRAM-ID.  HOSPEDIT.
000004      AUTHOR.  JON SAYLES.
000005      INSTALLATION. COBOL DEVELOPMENT CENTER.
000006      DATE-WRITTEN. 01/01/08.
000007      DATE-COMPILED. 01/01/08.
000008      SECURITY. NON-CONFIDENTIAL.
000009
000010      *****
000011      * A new comment ...
000012      * Jon's new comment
000013      ENVIRONMENT DIVISION.
000014      CONFIGURATION SECTION.
000015      SOURCE-COMPUTER. IBM-390.
000016      OBJECT-COMPUTER. IBM-390.
000017      INPUT-OUTPUT SECTION.
Command ==>
F1=Help    F2=Split   F3=Exit    F5=Rfind   F6=Rchange  F7=Up
F8=Down    F9=Swap    F10=Left   F11=Right  F12=Cancel
Scroll ==> PAGE

```

Multiple Edit windows are possible - but limited

- ISPF split-screen mode allows this ... but it is far from sufficient for complex, multi-module application programming problems



```

File Edit Edit_Settings Menu Utilities Compilers Test Help
-----
EDIT      DDS0001.TEST.COBOL(HOSPEDIT) - 27.26          Columns 00001 00072
***** ***** Top of Data *****
000001 *****
000002      IDENTIFICATION DIVISION.
000003      PROGRAM-ID.  HOSPEDIT.
000004      AUTHOR.  JON SAYLES.
Command ==> _____ Scroll ==> PAGE
F1=Help      F2=Split      F3=Exit      F5=Rfind      F6=Rchange      F7=Up
F8=Down      F9=Swap      F10=Left     F11=Right     F12=Cancel
. . . . .
File Edit Edit_Settings Menu Utilities Compilers Test Help
-----
EDIT      DDS0001.TEST.COBOL(HOSPCALC) - 06.05          Columns 00001 00072
***** ***** Top of Data *****
000001      IDENTIFICATION DIVISION.
000002      PROGRAM-ID.  HOSPCALC.
000003      AUTHOR.  JON SAYLES.
000004      ENVIRONMENT DIVISION.
Command ==> _____ Scroll ==> PAGE
F1=Help      F2=Split      F3=Exit      F5=Rfind      F6=Rchange      F7=Up
F8=Down      F9=Swap      F10=Left     F11=Right     F12=Cancel

```

And it's not just the basic tools ...

- Existing systems have grown by evolution over many years
- Many documented (and un-documented) dependencies
- Sheer volume of applications
 - thousands of batch jobs
 - thousands of programs
 - billions of lines of COBOL code run daily, not to mention on other schedules (e.g. end of quarter, end of year).
- Online transaction processing also factors in
- Without application analysis tools, teams have difficulty understanding even where to start

Agenda

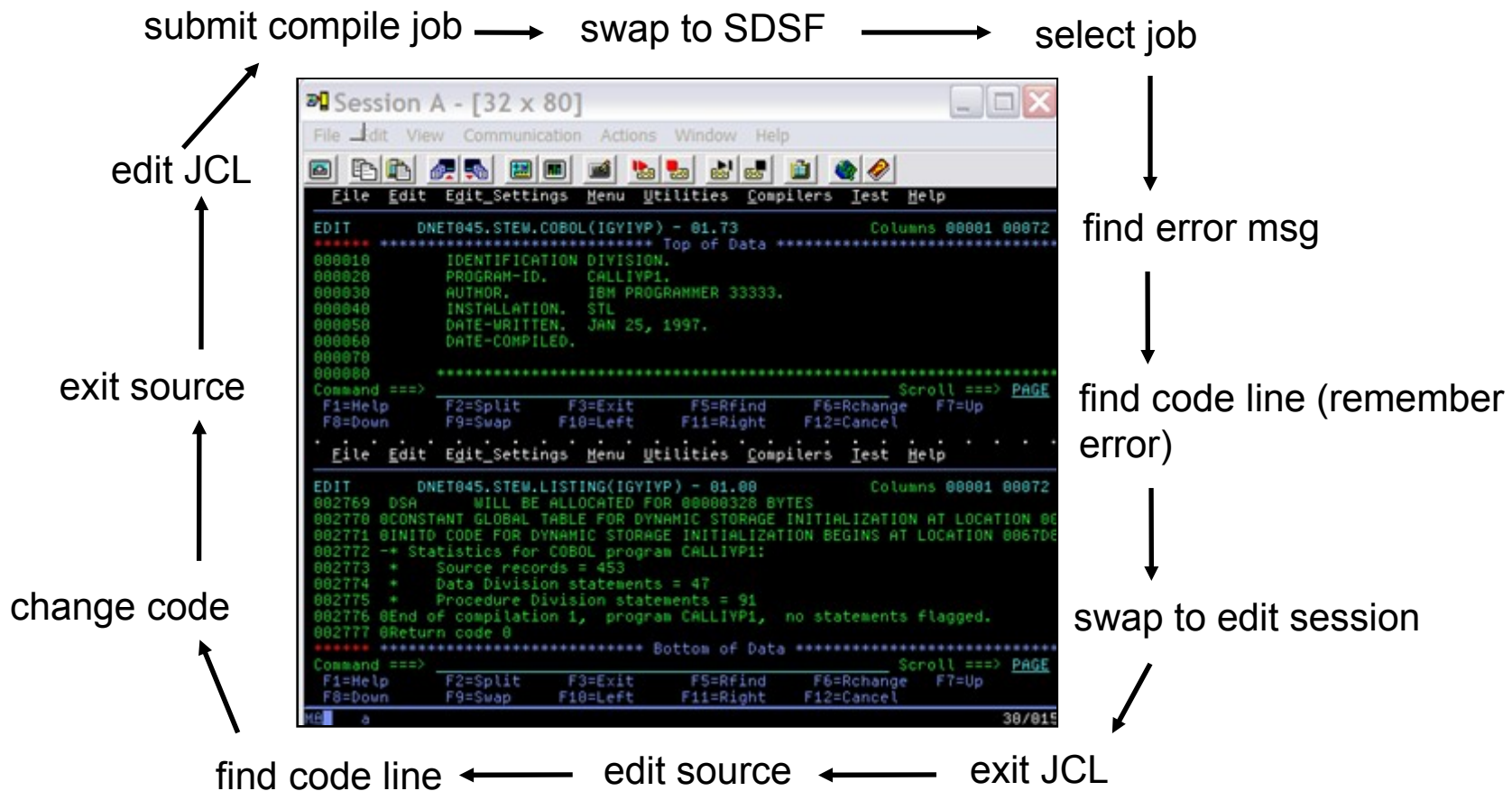
- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- Continual Discovery
- Reprise: Application Development is Hard

Rational Developer for System z and zEnterprise

- Practitioner tools for application development and enhancement
 - ▶ Java
 - ▶ COBOL
 - ▶ PL/I
 - ▶ C/C++
 - ▶ Assembler
 - ▶ JCL
- Supporting tasks of
 - ▶ Remote access to files and jobs
 - ▶ Analyze, Understand, Edit, Build, and Unit Test of applications
 - ▶ Remote interactive debug of applications running in multiple environments
 - ▶ Integration with SCMs including Team Concert and Endeavor
- Support for several source code location models
 - ▶ “remote” source code (source code held on development system)
 - ▶ “local” source code (source code held on system where IDE is running)

ISPF-based development

- Multiple screens/sessions and multiple disparate tools
- 20 x 80 characters of content



IDE-based development

- Common development environment for COBOL, PL/I, C/C++, and Java
- Simplified development with more information at your fingertips

The screenshot shows the IBM Rational IDE interface for z/OS. The main editor window displays a COBOL source file named REGIOA.cbl. The code includes a DISPLAY statement on line 35, which is highlighted with a red box. A red arrow points from the 'Edit Source' callout to this box. Below the editor, the 'Problems' view shows a list of errors, with one error highlighted: 'IGYPS2072-S "DISPLAI" was invalid. Skipped to the next verb, period or procedure-name d'. A red arrow points from the 'Double-Click on the Error' callout to this entry. On the left, the 'Outline' view shows the project structure, with 'REGIOA.cbl' and '010-INITIALIZATION.' highlighted by red boxes. Red arrows point from the 'Syntax Check' and 'Submit jobs, access job output, or open source members with a single click' callouts to these boxes. Another red arrow points from the 'Open and edit multiple source and JCL members simultaneously' callout to the project tree. A final red arrow points from the 'Error list in Problems view' callout to the error list.

Edit Source

Syntax Check

Submit jobs, access job output, or open source members with a single click

Open and edit multiple source and JCL members simultaneously

Statement in error indicated in source

Double-Click on the Error

Error list in Problems view

IDE-based Development

The screenshot displays the IBM Rational Developer for System z IDE interface. The main editor window shows the COBOL source file 'COBDATE.cbl' with the following code:

```

Line 16      Column 1      Insert
-----*A-1-B-----2-----3-----4-----5-----6-----
WORKING-STORAGE SECTION.
01  ChrDate.
   05  ChrDate-Length      pic s9(4) comp value 10.
   05  ChrDate-String      pic x(10) .
01  PicStr.
   05  PicStr-Length       pic s9(4) comp.
   05  PicStr-String       pic x(80) .
77  Lilian                 pic s9(9) comp.
77  Formatted-Date         pic x(80) .

PROCEDURE DIVISION.
*
MAIN-LOGIC.
  DISPLAY "Starting COBDATE".
  DISPLAY "-----"
  ACCEPT ChrDate-String from DATE.
    
```

The left-hand pane shows a project tree for 'z/OS Projects' containing various COBOL and C samples. The bottom-left pane shows the 'Properties' view for the 'PROGRAM: COBDATE', detailing its structure:

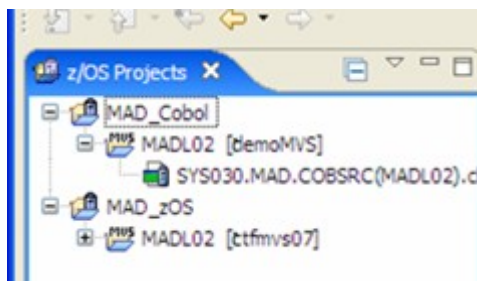
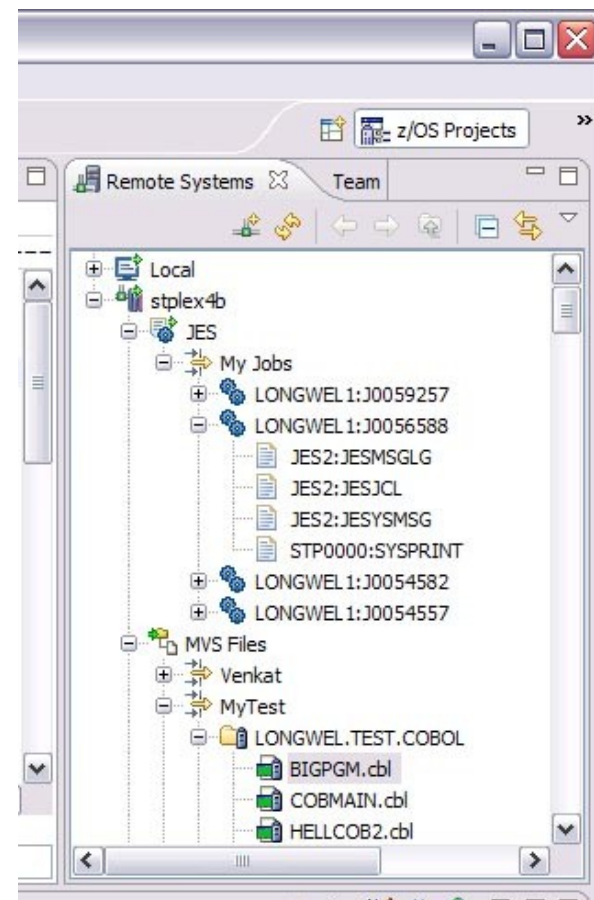
- PROGRAM: COBDATE
 - IDENTIFICATION DIVISION.
 - ENVIRONMENT DIVISION.
 - CONFIGURATION SECTION.
 - DATA DIVISION.
 - WORKING-STORAGE SECTION.
 - 01 ChrDate.
 - 01 PicStr.
 - 77 Lilian
 - 77 Formatted-Date
 - PROCEDURE DIVISION.
 - MAIN-LOGIC.

The bottom-right pane shows the 'Remote Error List' with a search for '*0C4*'. The results show an error with the following explanation:

Explanation	Results
0C4	Explanation: A program interruption occurred, but no routine had been specified to handle this type of interruption. Refer to the instruction description in Principles of Operation to find out how the instruction stops processing for the error condition.

Navigate datasets or jobs live on z/OS

- Connect to multiple hosts concurrently
- Respects existing security configurations and user IDs
- Search, filter, browse, edit, compare, migrate, and allocate new MVS datasets and USS files
- Copy source code, members, or datasets between systems with a few mouse clicks.
- Access JES queues submit jobs, view job state, and open output spools
- Submit TSO or USS commands
- Add datasets and members into projects to group applications and work items together logically
- Open an emulator in the IDE to configured hosts



Edit and syntax check source code

- Use advanced editing technology to:
 - ▶ Work with multiple source and JCL members concurrently from different systems
 - ▶ Perform ISPF-like commands in the workstation editor (e.g, FIND, CHANGE, INSERT LINE, etc)
 - ▶ Use syntax highlighting and code-completion to gain insight into available variables, verbs, and keywords
 - ▶ Quickly create programs from code templates, pattern definitions, or UML
 - ▶ Verify COBOL syntax with feedback as you type in real-time
- Issue syntax check commands against project source code
 - ▶ Syntax check remotely to ensure proper code structure before compilation
 - ▶ Syntax check locally ensure proper code structure and reduce server usage. RDz will download code and dependencies (e.g., copybooks) to the workstation as necessary
 - ▶ Syntax Errors are listed in the Remote error list. Double-click on the error to open the dataset and move to the line where the error was found

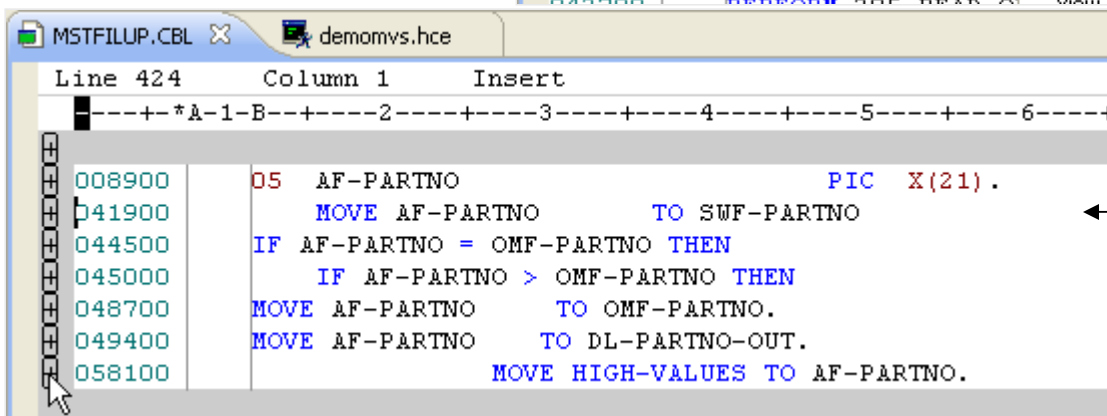
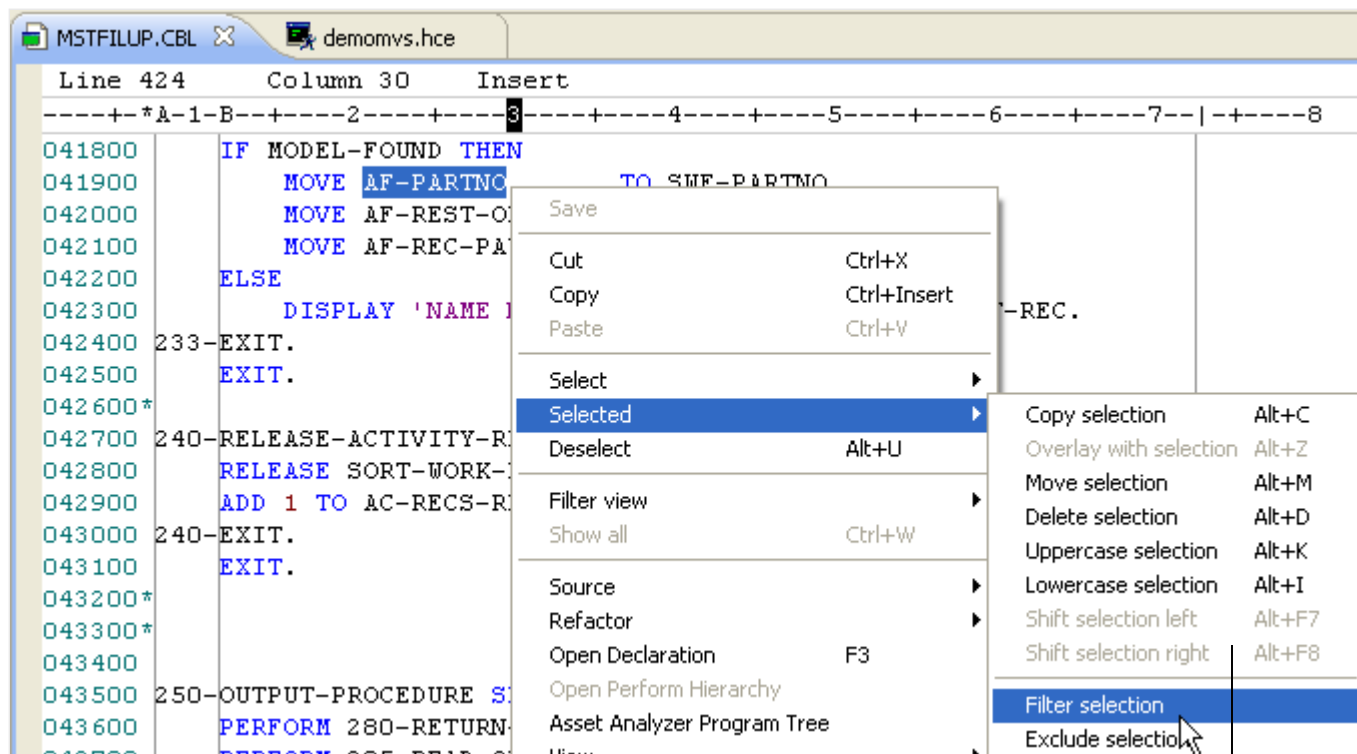
```

Row 37      Column 20      2 changes.
-----*A-1-B-----2-----3-----4-----5-----6-----
000032      PROCEDURE DIVISION.
000033      010-INITIALIZATION.
000034      * Initialize Program-work-fields, Program-flags,
000035      DISPLAY "Program REGIOA STARTING "
000036      MOVE 2 TO BRANCHFLAG.
000037      move
000038
000039
000040
000041
000042
000043      020-LOOP
000044
000045
000046
000047      divide value1 BY received-from-called GIVING
  
```

ID	Message
IGYDS1102	IGYDS1102-E Expected "DIVISION", but found "DIVISIO". "DIVISION" was assumed before "...
IGYDS1089	IGYDS1089-S "DIVISIO" was invalid. Scanning was resumed at the next area "A" item, level-nu...
IGYDS1082	IGYDS1082-E A period was required. A period was assumed before "DIVISIO".

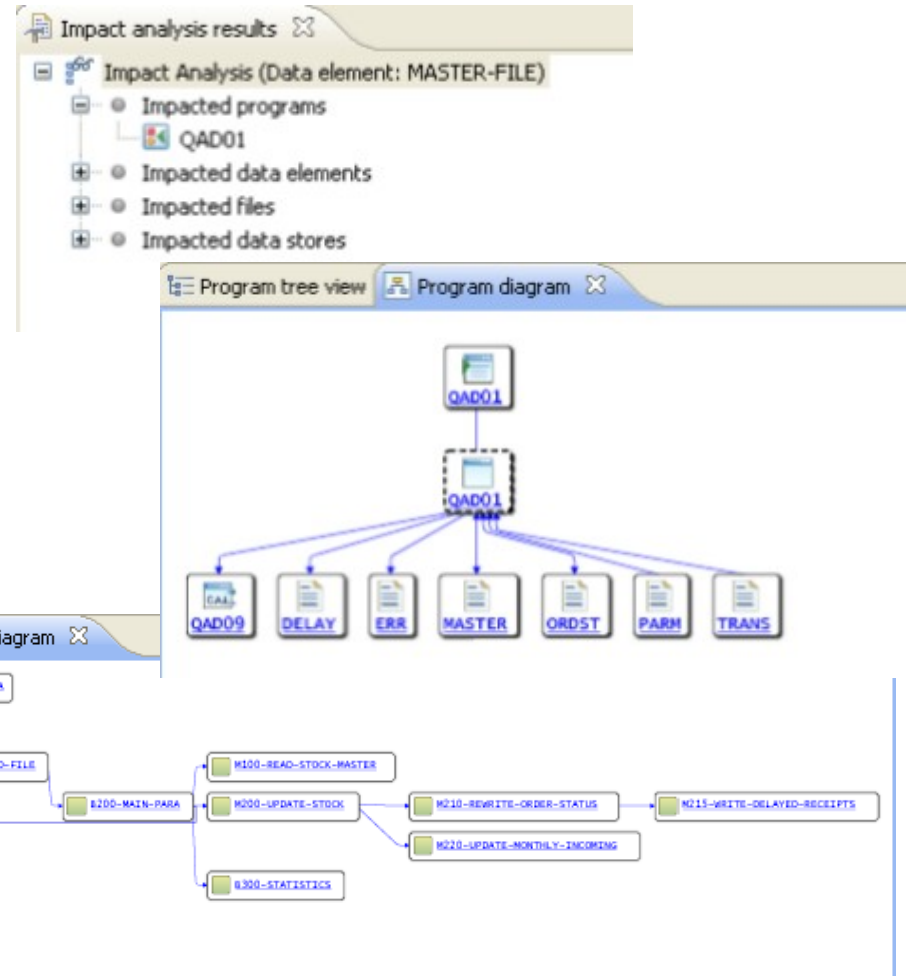
Isolating Code Elements

- Isolate and filter source code
- Multiple editor styles
 - ▶ “Eclipse”-style
 - ▶ ISPF-style
- Bookmark, and/or Expand & Collapse code details



Analyze applications using graphical diagrams

- Bring application analysis information into the IDE to aid in program development and understanding
 - ▶ Link code to data and runtime resources
 - ▶ Visualize code structure and flow
- Understand the effect of changes made in the IDE when deployed into production
 - ▶ Run impact analysis on code to determine affected modules
 - ▶ Size testing efforts and create workspaces for changes



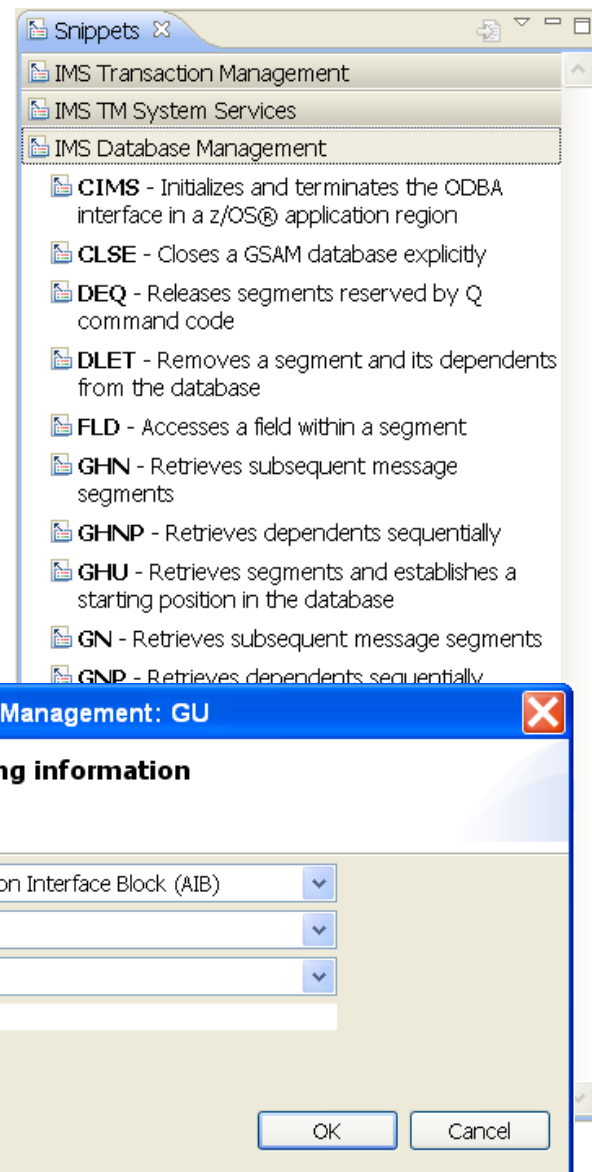
Program tree view | Data element table

Name	Level	Type
DELAY-FILE	0	FD
DELAY-STA		RAA Details
ERROR-DAT		View source (default)
ERROR-DES		References
ERROR-FIL		Modifications
ERROR-REC		References and Modifications
ERROR-STA		Impact Analysis
FILLER		
INP-DELAY-RCPT-REC	1	GRP
INP-DRCP-EXPECTED-DT	5	CHAR
IMP-DRCP-EXPECTED-DT	5	CHAR
IMP-DRCP-EXPECTED-DT	5	CHAR

NOTE: Features on this page require usage of Rational Asset Analyzer in conjunction with Rational Developer for System z or zEnterprise

Speed Development with Code generation

- Model Driven Development
 - use UML to generate COBOL code
- CICS
 - Create working CICS-DB2 CRUD transactions
- IMS statement insertion
 - 71 IMS code generation wizards aid to create IMS COBOL code inline
- DB2
 - Stored Procedure wizards
- Batch applications
 - VSAM / QSAM access program creation
 - Pattern-based code creation preview



Interactive test of database queries

The screenshot displays the IBM Data Studio interface with four main components:

- Left Panel (cursravg.cbl):** A COBOL program with a highlighted SQL block:

```
EXEC SQL
  DECLARE C1 CURSOR FOR
  SELECT DEPT, MIN(PERF), MAX(PERF), AVG(PERF),
         MIN(HOURS), MAX(HOURS), AVG(HOURS)
  FROM EMPL, PAY
  WHERE EMPL.NBR = PAY.NBR
  GROUP BY DEPT
END-EXEC.
```
- Middle Panel (*Script1.sql):** A SQL script containing the same query:

```
1 SELECT DEPT,
2 MIN(PERF), MAX(PERF), AVG(PERF),
3 MIN(HOURS), MAX(HOURS), AVG(HOURS)
4 FROM EMPL, PAY
5 WHERE EMPL.NBR = PAY.NBR
6 GROUP BY DEPT
```
- Right Panel (EMPL):** A table editor showing employee data. A context menu is open over a row, with 'Insert Row' selected.

NBR [CHAR(2)]	LNAME [CHAR(10)]	FNAME [CHAR(8)]
01	LOWE	ROB
02	SHIELD	BROOKE
03	MOORE	ROGER
04	EASTWOOD	Ron
05	Lim	ZERO
06	BURNS	GEORGE
07	O'NEIL	RYAN
08	MARVIN	Vijay
09	LANCASTER	Leo
10	BLAIR	LINDA
- Bottom Panel (SQL Results):** A table showing the results of the SQL query.

Status	Operation	Date	DEPT	2	3	4	5	6	7
Warning	SELECT DEPT, MIN(PERF), MAX(PERF), AVG(PERF), MIN(HOURS), MAX(...)	10/24/09	1 ACC	2	3	2	15.99	26.75	21.3700
			2 FIN	2	4	3	8.89	32.45	22.6966
			3 MKT	1	3	1	6.11	32.41	17.2500
			4 R&D	1	1	1	43.59	43.59	43.5900
			5 NULL	NULL	NULL	NULL	67.82	67.82	67.8200

Use Case – Database Applications:

1. Copy/Paste your SQL Declare into a SQL Script and run it – verify results
2. Without doing any other navigation open a test table, and edit row values – or modify the statement (or both)
3. Re-run the SQL Script – verify results
4. Return to step 2 – repeat until satisfied with functionality

Edit a program and edit its copybooks (all at the same time)

The screenshot displays three windows in the IBM z/OS development environment:

- MSTFILUP.CBL**: A COBOL program listing with line numbers 340 to 670. A context menu is open over this window, listing various editing actions such as 'Save', 'Cut', 'Copy', 'Paste', 'Select', 'Deselect', 'Filter view', 'Source', 'Refactor', 'Open Declaration', 'Run As', 'Debug As', 'Profile As', 'Validate', 'Software Analyzer', 'Team', 'Compare With', 'Replace With', 'Start Flagging Changed Lines', 'Local Syntax Check', 'Browse Copy Member', 'Open Copy Member', and 'Content assist'.
- PERSON.CPY**: A copybook listing with line numbers 1 to 6. It contains COBOL code for a personnel database, including 'COPYLIB FOR PERSONNEL DATABASE' and 'RECORD LENGTH IS 80 BYTES'. It defines fields like PERSON-NUMBER, PERSON-NAME, PERSON-FIRST-NAME, PERSON-LAST-NAME, PERSON-STREET-ADDRESS, PERSON-CITY-ADDRESS, PERSON-STATE-ADDRESS, PERSON-SALARY, and FILLER.
- PERSNFIL.CPY**: Another copybook listing with line numbers 7 to 6. It defines fields like PERSON_LAST_NAME, PERSON_STREET_ADDR, PERSON_CITY_ADDR, PERSON_STATE_ADDR, and PERSON_SALARY with their respective data types and lengths.

Interactive Debug of applications running on z/OS

The screenshot displays the IBM Rational Developer for System z debug environment. The main window shows the source code for 'ENGLAND.COBLIST.LISTING(COBDATE)'. A breakpoint is set at line 35. The 'Variables' window shows the state of variables: PICSTR is 'WWWWWWWWWZ, MMMMMMMMZ DD, YYYY', LILIAN is '0000155928', and FORMATTED-DATE is 'SUNDAY, SEPTEMBER 13, 2009'. The 'Registers' window shows the state of General Purpose Registers (%GPR0-%GPR6). The console window shows the program's execution flow and the reason for stopping: 'Program was stopped due to line/statement breakpoint at statement 35.'

Name	Value
PICSTR	'WWWWWWWWWZ, MMMMMMMMZ DD, YYYY'
PICSTR-LENGTH	+00050
LILIAN	+0000155928
FORMATTED-DATE	'SUNDAY, SEPTEMBER 13, 2009'

Name	Value
%GPR0	342925A4
%GPR1	3420027E
%GPR2	000127FC
%GPR3	342003D0
%GPR4	34200038
%GPR5	342108E0
%GPR6	00000000

```

Line 35      Column 1      Insert      Browse
-----
28          Move 6          to PicStr-Length
29          Call "CEEDAYS" Using ChrDate, PicStr, Lillian, Omitted.
30
31          Move 'WWWWWWWWWZ, MMMMMMMMZ DD, YYYY ' to PicStr-String .
32          Move 50          to PicStr-Length .
33          Call "CEEDATE" USING Lillian, PicStr, Formatted-Date, Omitted.
34          DISPLAY "*****".
35          Display Formatted-Date .
36          DISPLAY "*****".
    
```

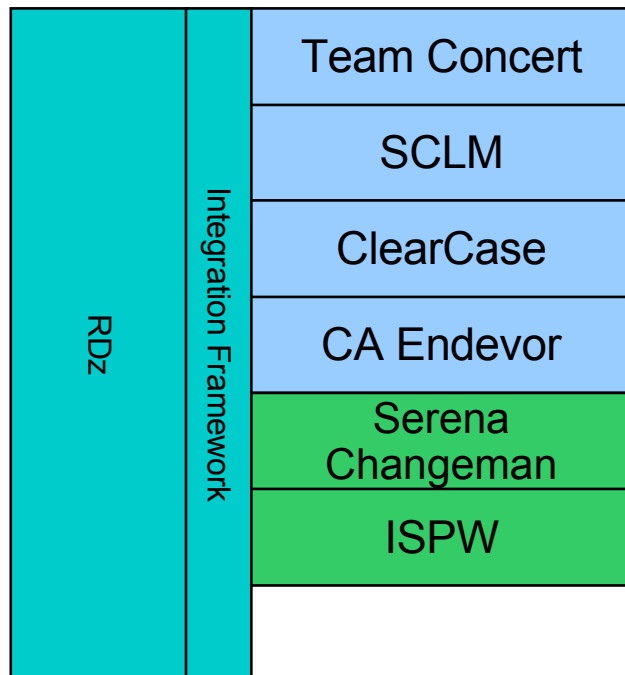
```

The subblocks in this program are nested as follows:
  1 COBDATE
Program was stopped due to line/statement breakpoint at statement 35.
    
```

Debug Engine Command: Enter Commands...

Access source code...

- RDz offers integration into a variety of Source Code Management (SCM) tools as well as a framework for creating SCM integration on your own
- Several vendors supply plug-ins to RDz to provide easy access to processes and source code controlled by their products



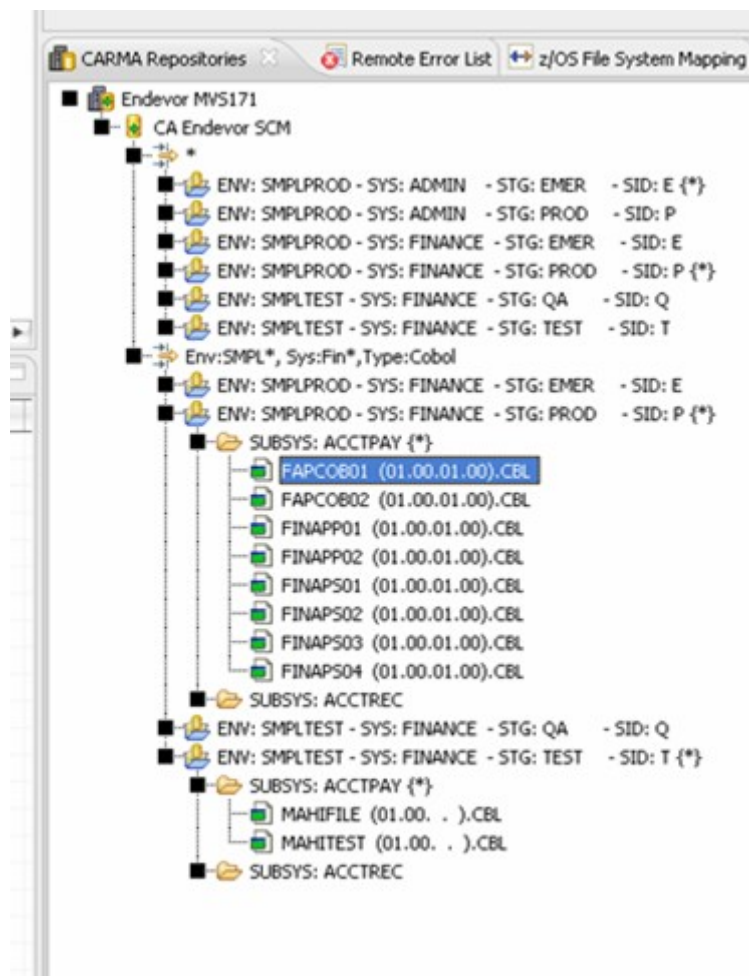
IBM Supplied

Vendor Supplied



Endevor Integration

- Filter and search through environments, systems, subsystems, members, and stages based on queries (equivalent to DISPLAY)
 - ▶ Filters saved across z/OS sessions
 - ▶ Easy access to common searches and members
 - ▶ Drill down into subsystems
- RETRIEVE members to z/OS projects
 - ▶ Access to typical RDz functionality like syntax check, content assist, debug, etc
- ADD/UPDATE members with single click
 - ▶ RDz remembers Endevor location for retrieve and adds back
- QuickEdit (browse) members from CARMA interface
- Integration with existing GENERATE configuration

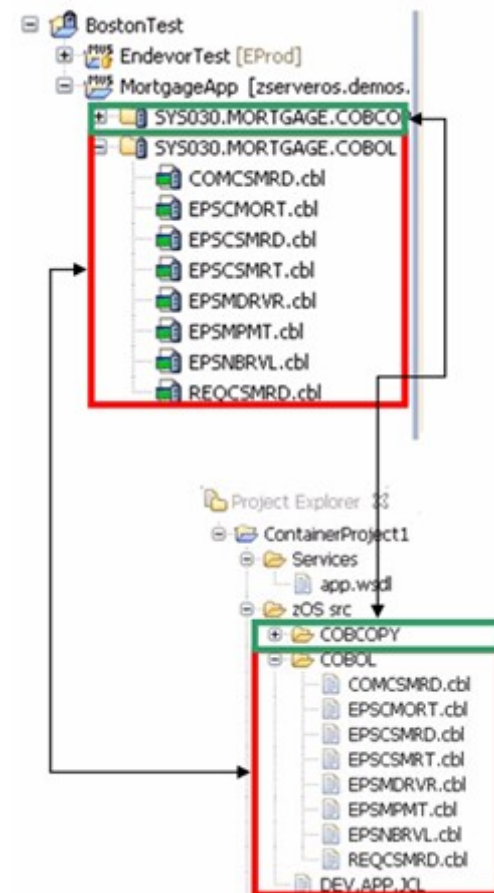


RTC integration with RDz - all tasks within a single IDE

- RTC provides
 - ▶ agility, collaboration and process
 - ▶ Work item planning and coordination
 - ▶ SCM and Build functions for z/OS (and other platforms)

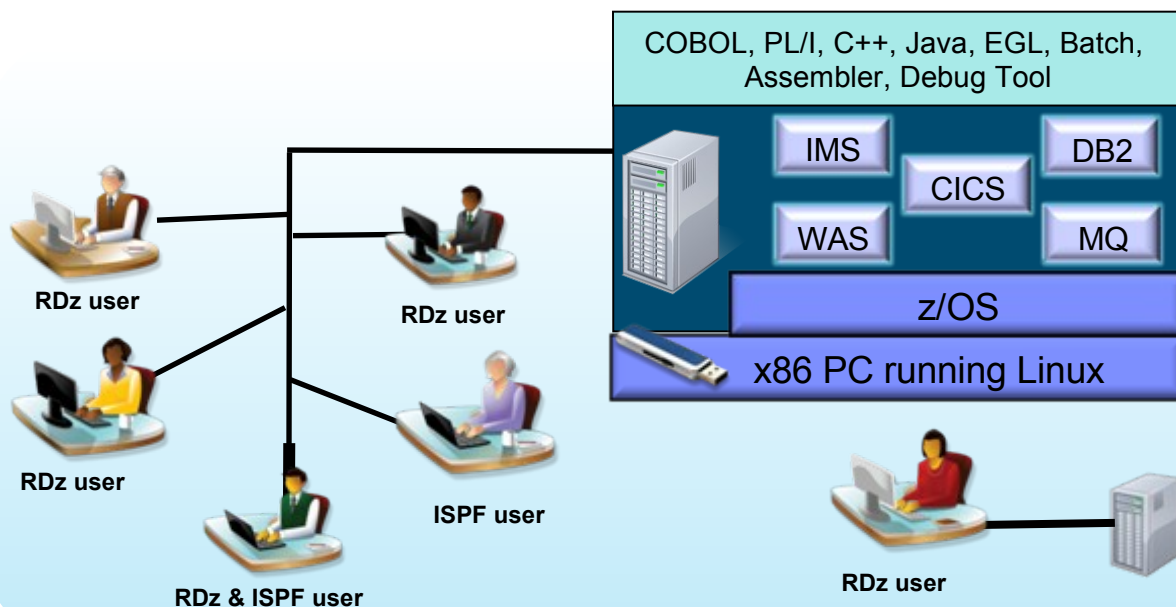
- RDz augments the development productivity & experience
 - ▶ files act as if on the host
 - ▶ Appropriate editors (COBOL, maps, etc.) and functions (content assist, syntax check, etc.)
 - ▶ High value functions (XML enablement, SFM, code generation from models, from UML, etc)

- RDz projects in RTC
 - ▶ RDz projects are a view into the RTC project
 - ▶ RDz projects provide a working set for the developer



RDz Unit Test Feature

Assisting application development for System z



- Liberate developers to rapidly prototype new applications
- Develop and test System z applications anywhere, anytime!
- Free up mainframe development systems for production capacity
- Eliminate costly delays by reducing dependencies on operations staff

For more information:
 Session 8369
 Monday – 4:30-5:30PM
 Room 201A

Note: This Program is licensed only for development and test of applications that run on IBM z/OS. The Program may not be used to run production workloads of any kind, nor more robust development workloads including without limitation production module builds, pre-production testing, stress testing, or performance testing.

Agenda

- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- Continual Discovery
- Reprise: Application Development is Hard

Using Tools - There is a Learning Curve

- It's true.
- Any new tool is, at first, overwhelming.

- ISPF and “green screen” users
 - Compatibility mode for editor environment
 - Remote system access
 - 3270 emulator included

- IDE-comfortable users
 - Multiple edit window support
 - multiple editor selections available based on preference
 - extensive preferences and customization
 - context menus for most common tasks

Agenda

- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- **Continual Discovery**
- Reprise: Application Development is Hard

Continual Discovery

- Challenge Yourself
 - Resolve to learn something new every day
- Find a Buddy
 - learn from what each of you have found
- Impress your Friends
 - find out something cool? Share you knowledge!
- Be Social!
 - Join a user group, discussion group, or online community
 - ask questions, or just lurk and learn
 - (See links at end of presentation)

Agenda

- Application Development is Hard
- Tools to the rescue!
- Using tools is Hard
- Continual Discovery
- Reprise: Application Development is **NOT SO** Hard

The hardest part of Application Development is ...

- Getting started

The hardest part of Application Development is ...

- Getting started ... and
- Staying engaged and on task

Application development for z/OS used to be like this ...



```

File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT      DDS0001.TEST.COBO(L(HOSPEDIT) - 27.26          Columns 00001 00072
***** ***** Top of Data *****
000001      *****
000002      IDENTIFICATION DIVISION.
000003      PROGRAM-ID.  HOSPEDIT.
000004      AUTHOR.  JON SAYLES.
000005      INSTALLATION. COBOL DEVELOPMENT CENTER.
000006      DATE-WRITTEN. 01/01/08.
000007      DATE-COMPILED. 01/01/08.
000008      SECURITY. NON-CONFIDENTIAL.
000009
000010      *****
000011      * A new comment ...
000012      * Jon's new comment
000013      ENVIRONMENT DIVISION.
000014      CONFIGURATION SECTION.
000015      SOURCE-COMPUTER. IBM-390.
000016      OBJECT-COMPUTER. IBM-390.
000017      INPUT-OUTPUT SECTION.
Command ==>
F1=Help    F2=Split    F3=Exit    F5=Rfind    F6=Rchange    F7=Up
F8=Down    F9=Swap     F10=Left   F11=Right   F12=Cancel
Scroll ==> PAGE

```

And it still can be ...

... but why do that when you can use all these features?

The screenshot displays the IBM Rational Developer for System z interface. The main editor window shows a COBOL program named COBDATE.cbl with the following code:

```

Line 16      Column 1      Insert
-----*A-1-B-----2-----3-----4-----5-----6-----
WORKING-STORAGE SECTION.
01  ChrDate.
   05  ChrDate-Length      pic s9(4) comp value 10.
   05  ChrDate-String      pic x(10) .
01  PicStr.
   05  PicStr-Length       pic s9(4) comp.
   05  PicStr-String       pic x(80) .
77  Lillian                pic s9(9) comp.
77  Formatted-Date        pic x(80) .

PROCEDURE DIVISION.
*
MAIN-LOGIC.
  DISPLAY "Starting COBDATE".
  DISPLAY "-----"
  ACCEPT ChrDate-String from DATE.
    
```

The interface includes a Project Explorer on the left showing a project structure with folders like 'BeerBottles' and 'COBOL-LE-DateTime [stplex4b]'. A Properties/Outline pane at the bottom left shows the program structure for COBDATE, including sections like IDENTIFICATION, ENVIRONMENT, CONFIGURATION, DATA, and PROCEDURE DIVISION.

At the bottom, the Remote Error List pane shows a search for '*0C4*' with the following results:

Explanation	Results
0C4	Explanation: A program interruption occurred, but no routine had been specified to handle this type of interruption. Refer to the instruction description in Principles of Operation to find out how the instruction stops processing for the error condition.

Rational Developer for System z and zEnterprise

- Eases developers into working with System z systems
 - **Easier to get started**

- Offers many features within the Integrated Development Environment
 - **Keeps developers on task and engaged**



www.ibm.com/software/rational

© Copyright IBM Corporation 2010,2011. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. IBM, the IBM logo, Rational, the Rational logo, Telelogic, the Telelogic logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

Useful Links

- Rational Developer for System z Information:
 - <http://www.ibm.com/software/rational/products/developer/systemz/>
- Rational Developer for System z Infocenter:
 - <http://publib.boulder.ibm.com/infocenter/ieduasst/rtnv1r0/index.jsp?topic=/com.ibm.iea.rdz/rdz/wdz76.html>
- Additional Video Demonstrations:
 - http://websphere.dfw.ibm.com/atdemo/atdemo_rdz.html
 - http://websphere.dfw.ibm.com/atdemo/atdemo_rdz_zosad_recorded.html
- Rational Developer for System z - Distance Learning:
 - <https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/Learn%20RDz-Learn%20COBOL>
- Rational “COBOL Cafe” online discussion group:
 - <https://www.ibm.com/developerworks/rational/community/cafe/cobol.html>
- Rational Developer for System z - RDz hub:
 - <https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=df67969e-ba40-44c7-a1ca-ef4a2aa99e01>
- My information
 - email: hahnt@us.ibm.com
 - Blog: <https://www.ibm.com/developerworks/mydeveloperworks/blogs/applicationmodernization>