



The Modern Mainframe – At the Heart of Your Business

Productive Tools for Application Development



ODI Needs Productive Developer Tools

We have a lot of new solutions to build. If I build them all on System z, I will have to find more System z development skills



**On Demand Insurance
CIO**

With modern tools like WebSphere Developer for zSeries, you can double the productivity of your existing team



IBM

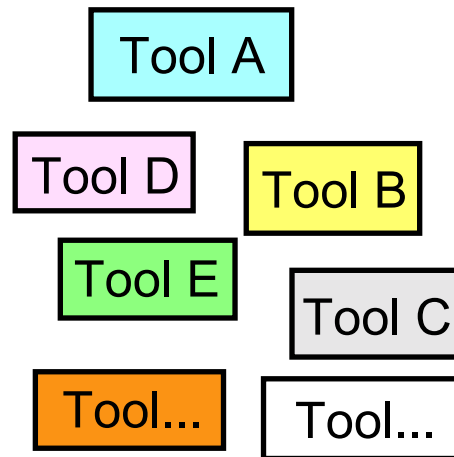
WebSphere Developer for System z (WDz)

- IBM's latest tool for System z development
 - ▶ Version 7 shipped Dec 2005
- Single integrated tool for developing mainframe applications using both traditional and newer technologies
 - ▶ Develop traditional COBOL/PL1/ASM/JCL/BMS/EGL applications
 - ▶ Develop Java/J2EE and Web applications, including JSF and struts
- Boosts developer productivity compared to green screen tools
 - ▶ Workstation tool based on Eclipse
- Leverage existing assets in a Service Oriented Architecture
 - ▶ Quickly expose CICS transactions as native services
 - ▶ Visually wire together CICS transactions into a business flow

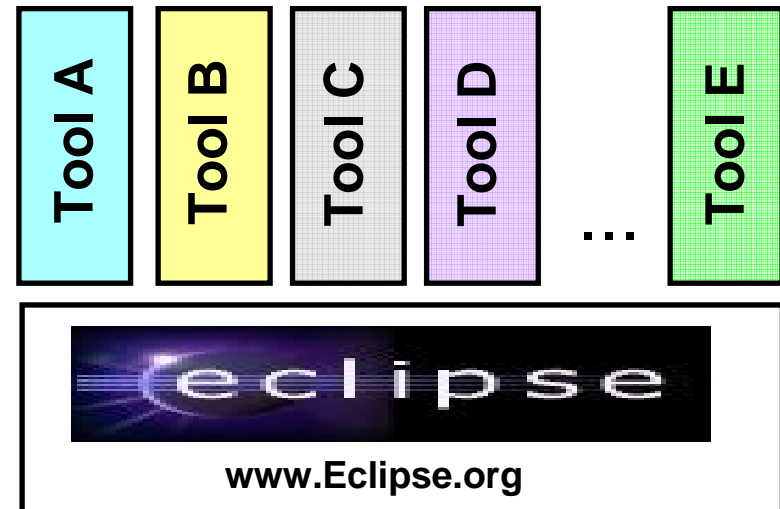
WDz is Built on Eclipse

Eclipse Provides an Integrated Toolset

Individual Tools



Integrated Tools



■ Challenges:

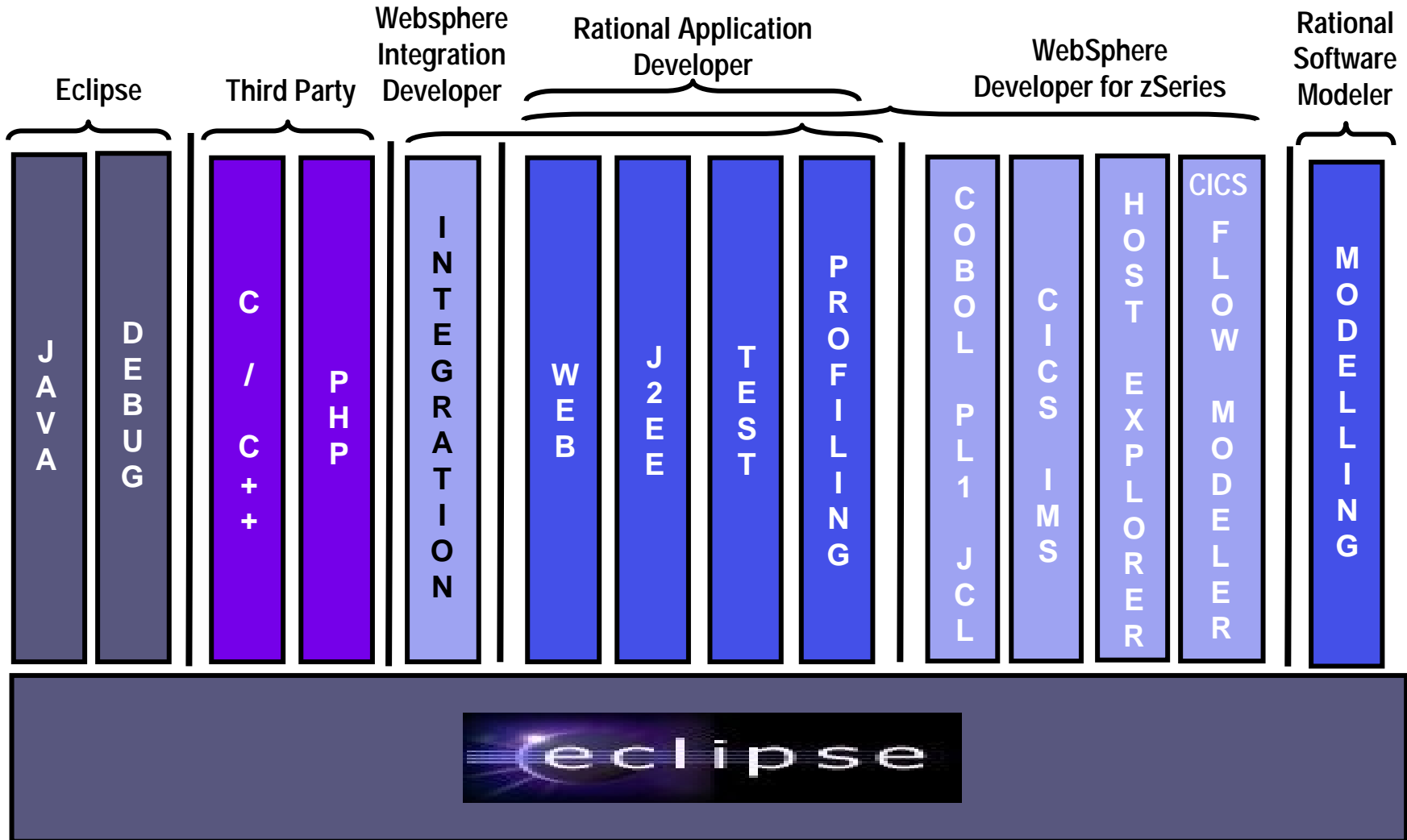
- ▶ Each task requires different tool
- ▶ Multiple repositories
- ▶ Multiple skills required for each tool
- ▶ Different look and feel
- ▶ Different semantics
- ▶ Difficult to manage
- ▶ Duplicate functions

■ Benefits:

- ▶ Tools integrated on open source platform
- ▶ Consistent User Interface
- ▶ Same semantics
- ▶ Supports multiple roles
- ▶ Common Repository between all tools
- ▶ Pluggable tools framework
- ▶ Integrated Test Environment

IBM Tools for SOA Development

Function delivered as perspectives in eclipse



WDz Highlights for Traditional Development

- Interactive, workstation-based environment
 - ▶ Interactive access to z/OS
 - ▶ Work with host files as though they are local

- Boosts developer productivity compared to “green screen” tools
 - ▶ Interactive edit/compile/debug on the workstation
 - Work offline or online
 - ▶ TSO commands, job generation, submission, remote debug to z/OS
 - ▶ Language sensitive editors

Interactive Access to z/OS

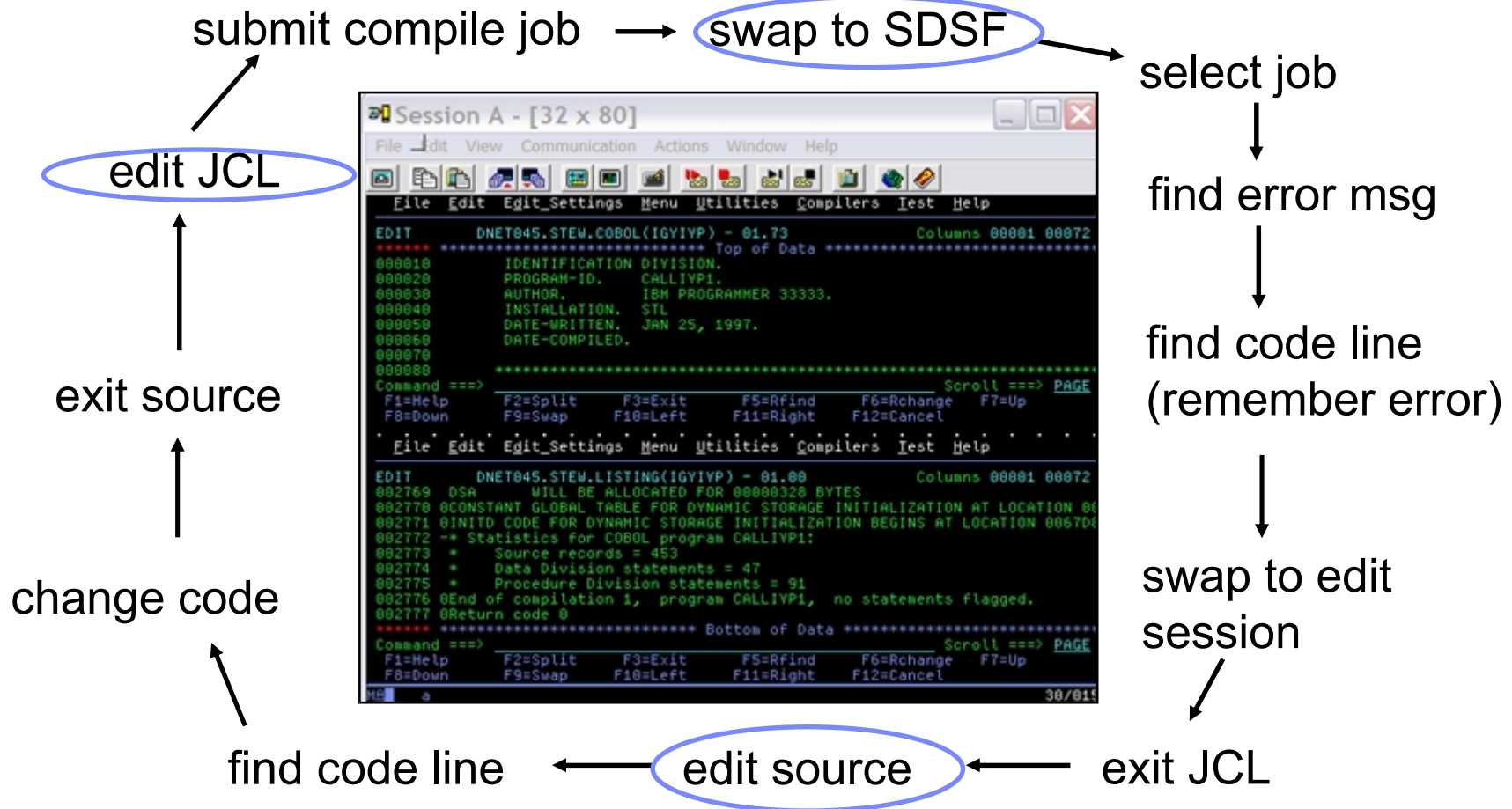
- Connect to z/OS system using Remote System Explorer
- Work with host libraries and datasets
- Job generation, submission, and monitoring
- TSO/USS command execution
- Mainframe integration:
 - ▶ Issue TSO commands or jobs from desktop
 - ▶ Full screen 3270 access
- Host Source Code Management (SCM) access through CARMA (Common Access Repository Manager)
 - ▶ Unified interface for accessing different host based SCM tools

DEMO: Introducing WebSphere Developer for zSeries

- Perspectives and views
- Remote System Explorer
- Working with host datasets
- Language-sensitive COBOL editor



ISPF: How Does Traditional “Green Screen” Development Work?



- Programmer goes through a sequence of screens in order to get the job done
 - ▶ 3.4 listings, job listings, SDSF outputs, etc.
- Programmer is constantly flipping back and forth between these ISPF screens

Simplified Development with WDz Perspectives

The screenshot displays the IBM Rational Software Development Platform interface for z/OS Projects. The main editor window shows the source code for REGIDA.cbl, with a red box highlighting a syntax error on line 35: "DISPLAI *Program REGIOA STARTING *". The error list in the Tasks view at the bottom shows the message: "IGYPS2072-S 'DISPLAI' was invalid. Skipped to the next verb, period or procedure-". The Outline view on the left shows the COBOL structure, with "010-INITIALIZATION." highlighted. Annotations include: "Syntax Check" pointing to the error list, "Edit source" pointing to the source code editor, "Statement in error" pointing to the highlighted line in the editor, "double click on the error" pointing to the error list entry, and "Error list in Tasks view" pointing to the error list. A note at the bottom left states "Outline view presents COBOL structure".

Syntax Check

Edit source

Statement in error

double click on the error

Error list in Tasks view

Outline view presents COBOL structure

Benefit: Simplified development for COBOL and PL/1 through intuitive, interactive screens

WDz Advantages Over Traditional Host-based Tools

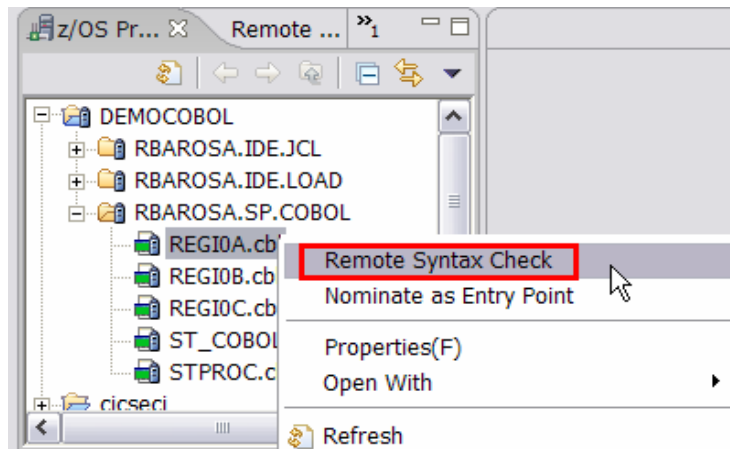
- Do multiple concurrent activities
 - ▶ With WDz, developer can multitask, all of the required perspectives are readily available
 - ▶ ISPF allows a maximum of 2 concurrent activities
 - No such limit in WDz
- Capabilities of a GUI editor
 - ▶ Color coding, Content assist, Outline view ..and many others
- Visual debugging
 - ▶ Identify problems quickly and intuitively

Productivity Features for COBOL, PL/1 Programmers: Content Assist

Content Assist completes keywords, statements and data names

Benefit: Developers complete code more accurately and efficiently

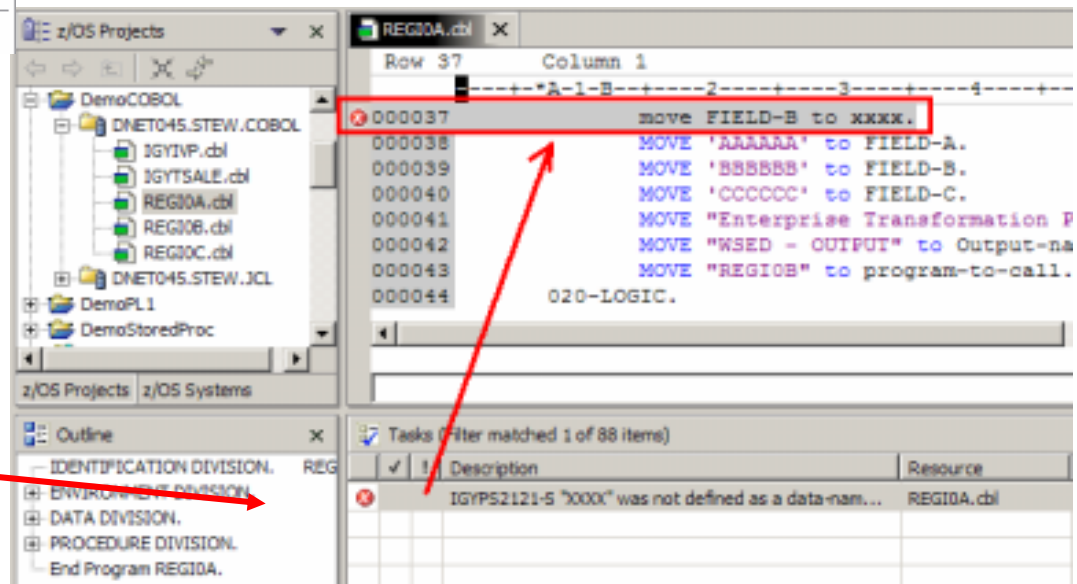
Productivity Features for COBOL, PL/1 Programmers: Syntax Checking



Do local or remote syntax checking..

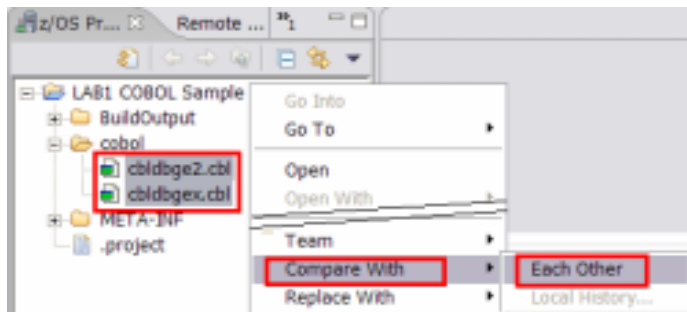
Benefit: Developers don't have to wait till compile time to check syntax

Just double-click to find the error



Productivity Features for COBOL, PL/1 Programmers: Compare Two Files

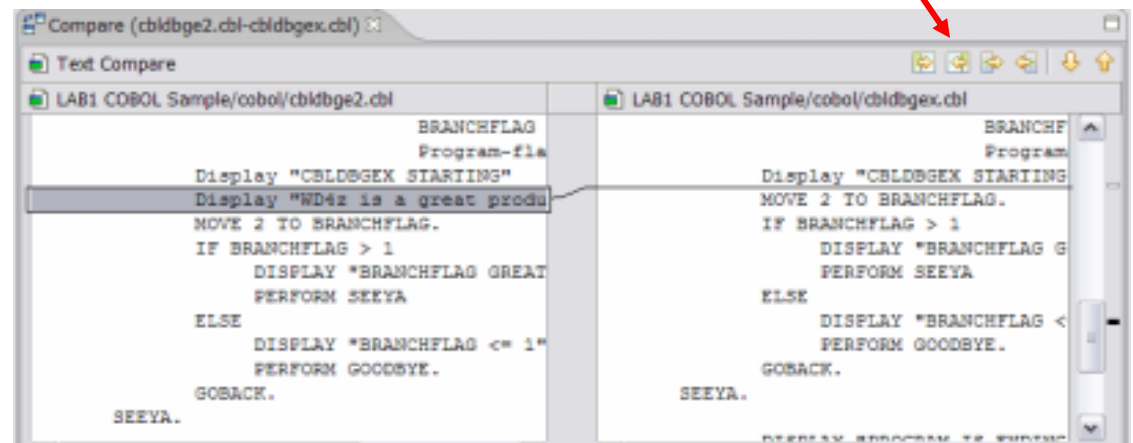
Underlying Eclipse Productivity Features are Available for z/OS Assets...



Navigate through and merge differences using icons

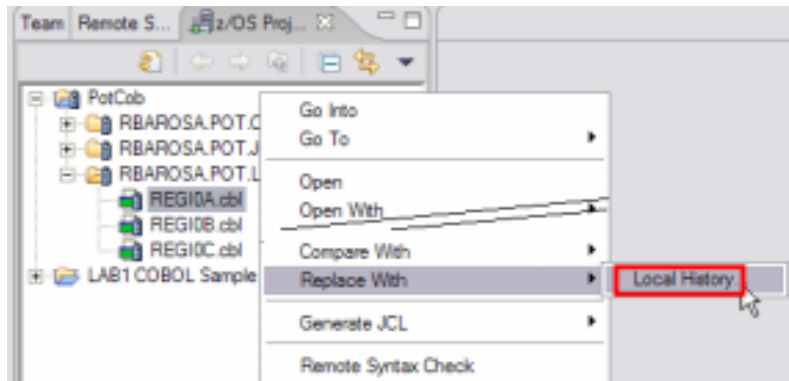


Benefit: Productivity
maintaining COBOL/PL1/JCL
files



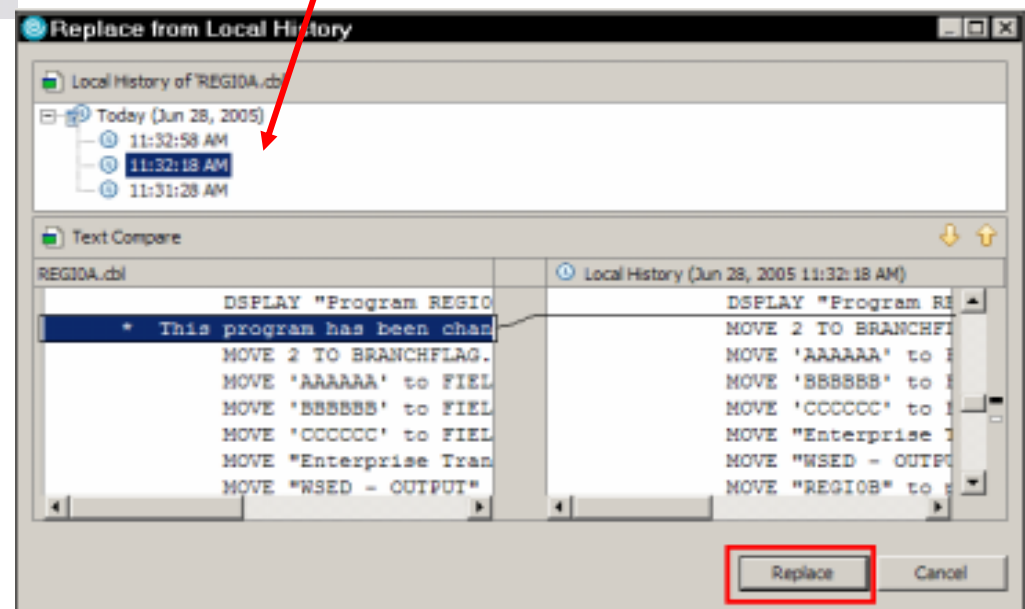
Productivity Features for COBOL, PL/1 Programmers: Replace with Local History

Underlying Eclipse Productivity Features are Available for z/OS Assets...



Keep as many local versions as you want and compare with the z/OS current version..

Benefit: Productivity maintaining COBOL/PL1/JCL files



DEMO: WebSphere Developer for zSeries Productivity Features

- Content assist
- Syntax checking
- Compare two files
- Local history



Productivity Features for COBOL, PL/1 Programmers: Generate JCL and Submit to z/OS

JCL generated from COBOL or PL/1 Code

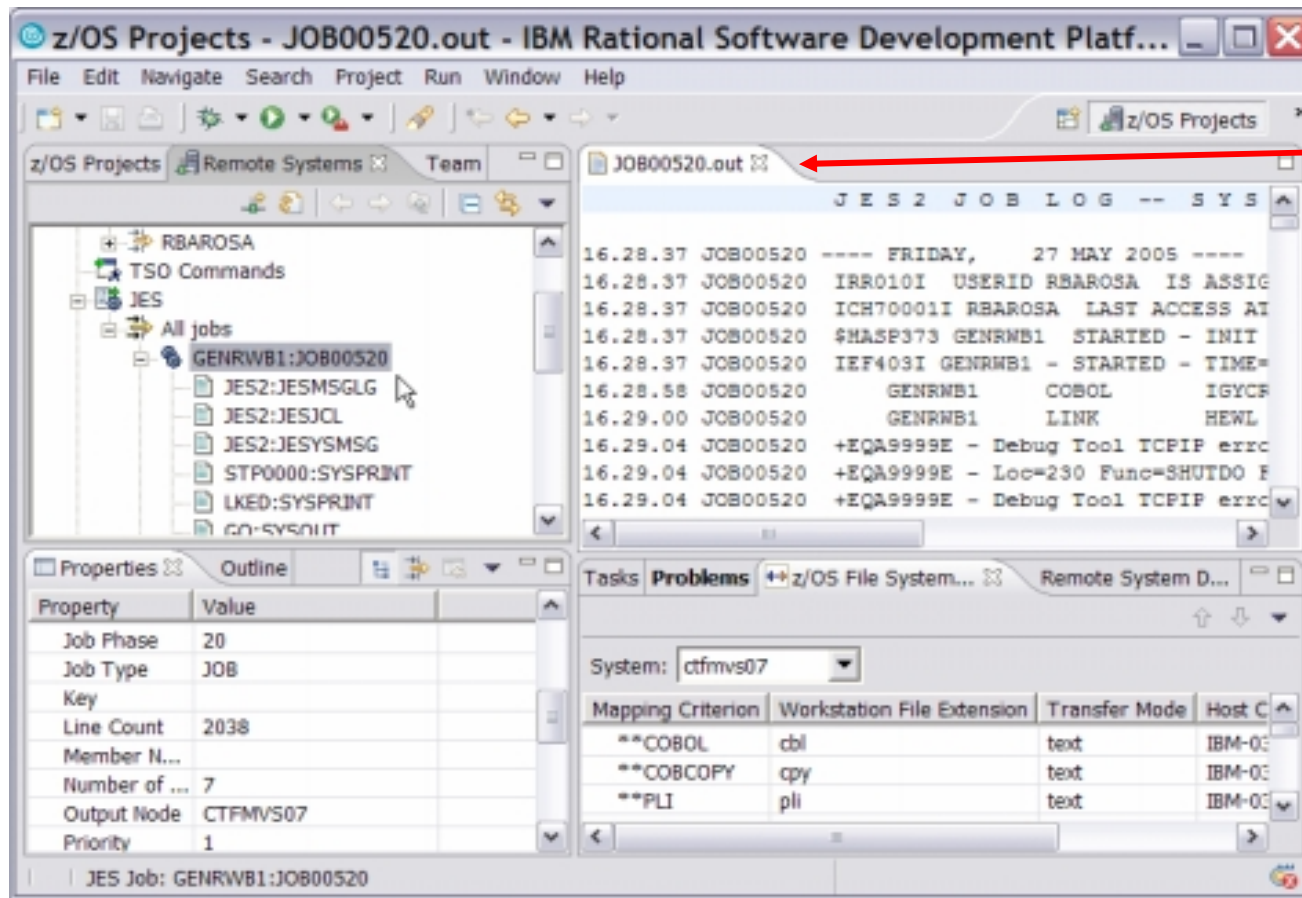
The screenshot displays the z/OS IDE interface. On the left, the 'z/OS Projects' tree shows a project named 'DNET045.STEW.COBOL' containing COBOL source files (IGYIVP.cbl, IGYTSALE.cbl, REGIOA.cbl, REGIOB.cbl, REGIOC.cbl) and a 'DNET045.STEW.JCL' folder containing JCL files (DEBUG.jcl, IBMZJSO1.jcl, IGYIVP.jcl, and REGIOA.jcl). The 'REGIOA.jcl' file is selected and circled in blue. A blue arrow points from the text 'JCL generated from COBOL or PL/1 Code' to this file. The main editor window shows the content of 'REGIOA.jcl' with the following code:

```
Row 1      Column 1
000001 //REGED511 JOB ,
000002 //  MSGCLASS=H,TIME=(,4),REGION=28M,COND=(16,LT)
000003 //  JCLLIB ORDER=DNET045.WSED511.JCL
000004 //*
000005 //DELLIST EXEC PGM=IDCAM5
000006 //SYSPRINT DD SYSOUT=*
000007 IF LASTCC = 8 THEN SET MAXCC = 4
000008 DELETE DNET045.IDECC
000009 IF LASTCC = 8 THEN S
000010 /*
000011 //STP0000 EXEC PROC=EL
```

Below the editor is an 'Outline' window showing a summary of the JCL code, including job names and execution procedures. A 'Tasks' window is also visible. On the right, a context menu is open over the 'REGIOA.jcl' file in the project tree, with the 'Submit' option highlighted in red.

Benefit: Developers focused on business logic and not on writing JCL

Productivity Features for COBOL, PL/1 Programmers: Monitoring Job Output



Output can be seen like local files.. Can use Find, etc..

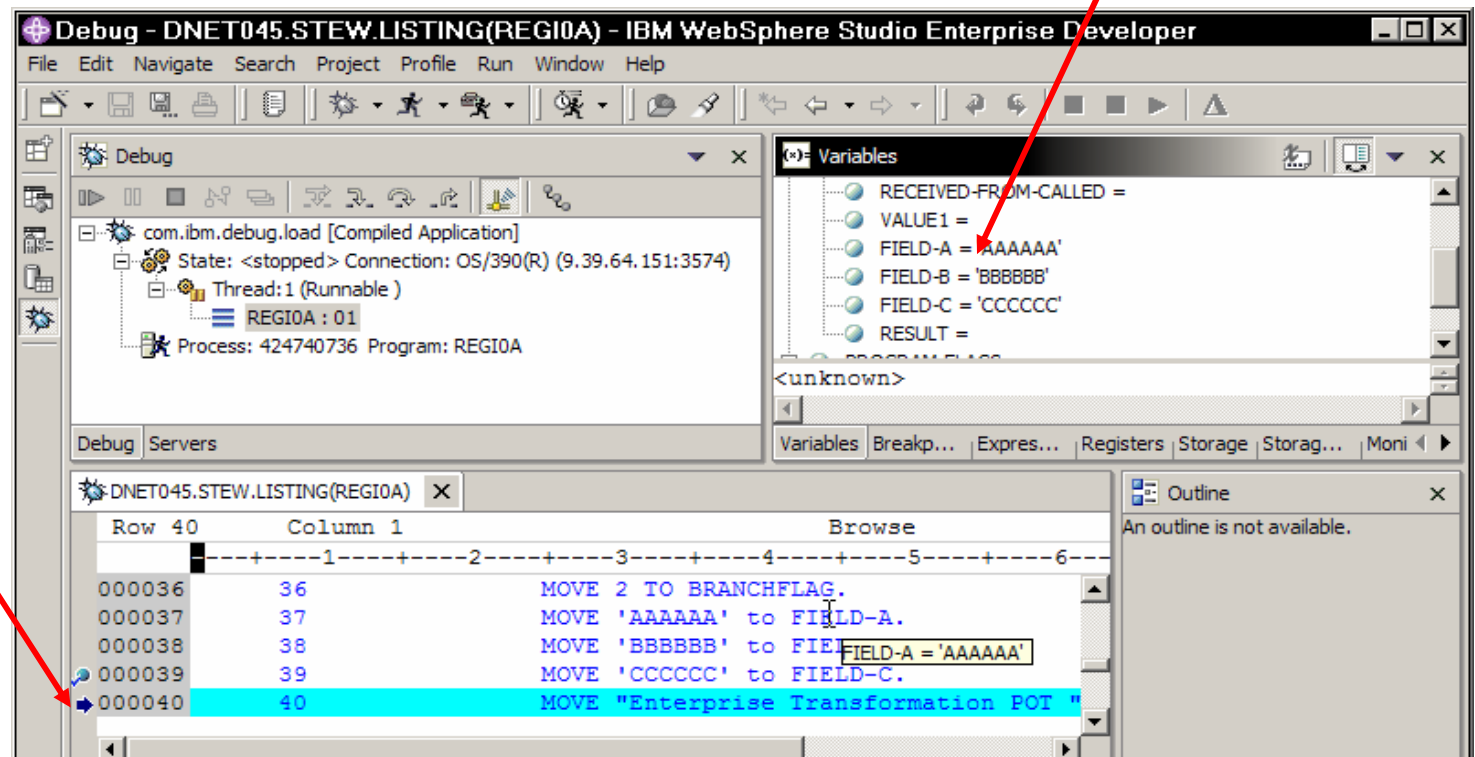
Benefit: Developers do not have to continually switch between systems to use SDSF. No TSO need.

Productivity Features for COBOL, PL/1 Programmers: Remote/Local Debug

- Same Debug Perspective used for COBOL, PL/1 & Java, etc..
- WDz is the ONLY product in the market that does END-to-END debug.. you can debug HTML --> JSP --> Java --> COBOL on z/OS... in the same IDE

Change contents, etc..

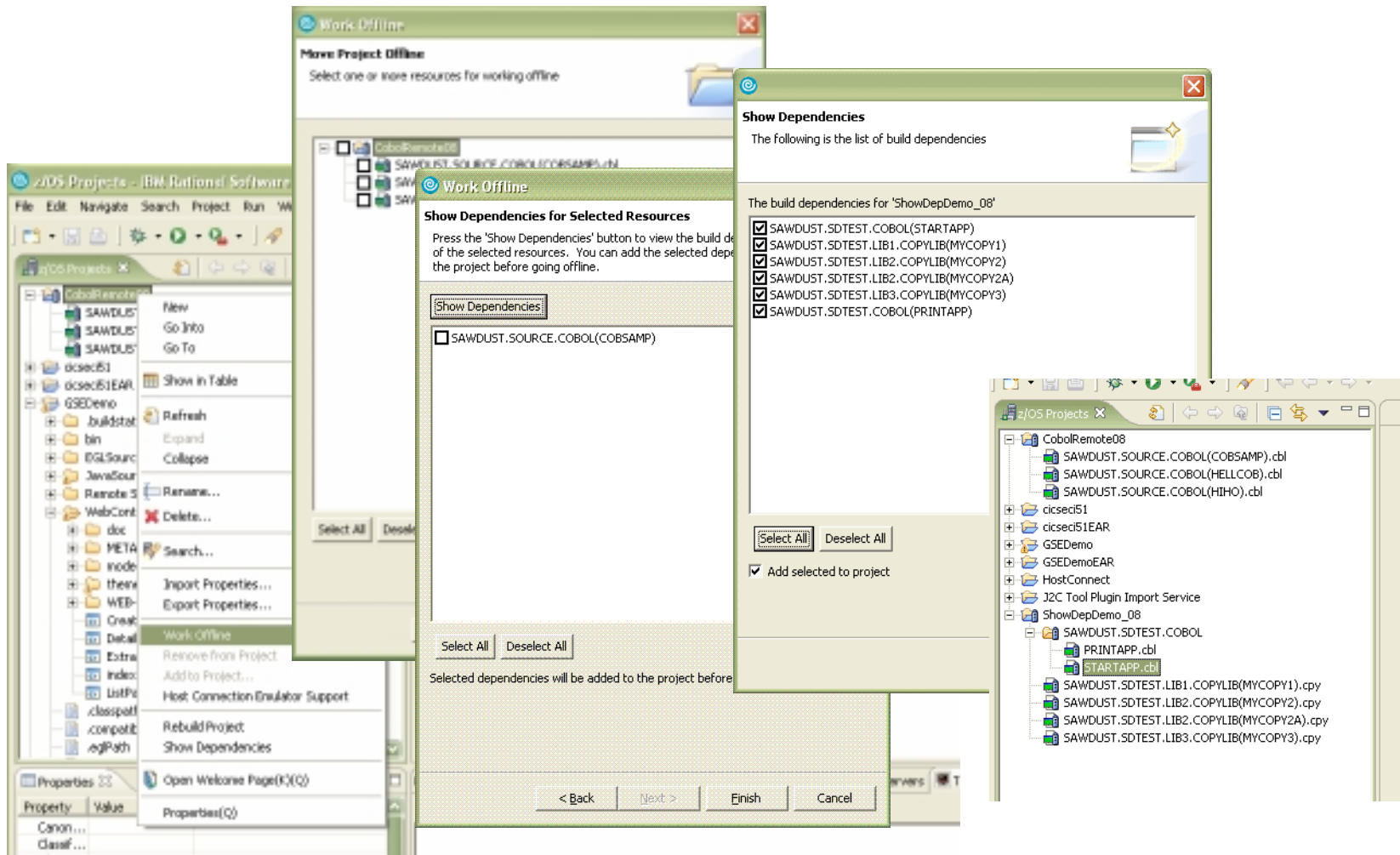
Breakpoints, watchpoints, Jump to, Run to etc..



Benefit: Consistent debugging environment for COBOL, PL/1, Java

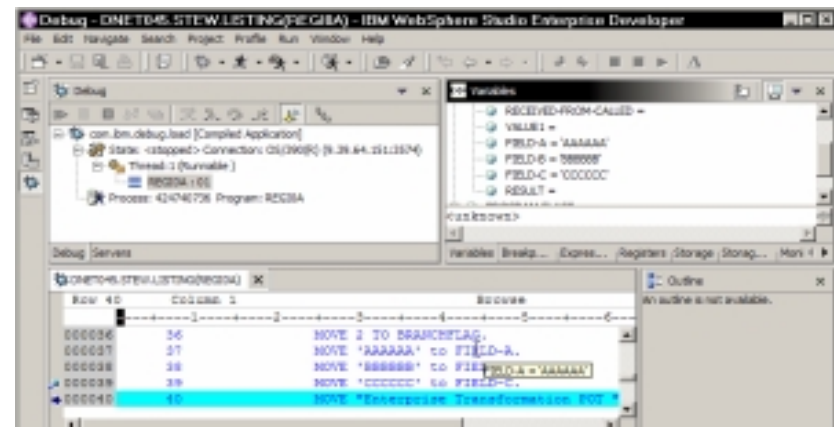
Enhanced Offline Capabilities

- Work offline or online
- Show dependencies
- Wizard to bring all dependent files that need to be in the project



DEMO: Developing a Host Application Using WDz

- Edit/Compile/Debug
- Show use of WDz productivity features
 - ▶ Write COBOL code, use Content Assist, syntax Check
 - ▶ Generate JCL for Compile-Link-Go, submit
 - ▶ Use Visual Debugger to walk through code as it executes



Productive Developer Tools for SOA Solutions

How about support for newer technologies like Web pages, Web Services and SOA?



**On Demand Insurance
CIO**

Remember, we already used WebSphere Developer for zSeries to help build parts of that external broker solution. Let's review what it can do.



IBM

WDz Highlights for Web and SOA Solutions

- SOA and Web Services Development
 - ▶ Quickly expose CICS transactions as services
 - XML Services for the Enterprise
 - ▶ Visually wire together CICS transactions into a business flow
 - Service Flow Modeler

- Web UI Development
 - ▶ Intuitive, drag-and-drop visual construction of user interfaces based on open standards
 - JSF, SDO and Struts

Web UI Development

- Leverage the full web development capabilities of Rational Application Developer
 - ▶ Tools to develop, test and deploy standards-based web applications

- Visual, interactive Web development
 - ▶ Develop Static and Dynamic Web pages
 - HTML, Java Server Pages, J2EE
 - Templates, Stylesheets, Web Diagrams
 - ▶ Java Server Faces (JSF)
 - Drag and drop User Interface components onto web page
 - ▶ Struts
 - Implement a model-view-controller design for web applications

- Enterprise Generation Language (EGL) Java/Web development
 - ▶ Code in higher level language, then generate to language of choice (Java or COBOL)
 - Tight integration with Java Server Faces

WebSphere Developer for zSeries Dramatically Lowers the Cost of Development for System z

- Productivity Increases over ISPF:
 - ▶ COBOL and PL/I Traditional Development
 - + 5% in program understanding
 - +10% in edit activities
 - +44% in compile/debug activities
 - ▶ Web Services and SOA development
 - +300% in programmer productivity
- Workstation based edit, compile, debug saves mainframe cycles
 - 80% reduction in cost of compiles

Based on customer studies at
Fiducia, Deutsche Bank AG,
Nationwide, etc.



**NEW! An
independent
study from the
Branham Group**

Discover Your Assets on the Mainframe

How do new programmers discover existing assets?
If they make changes, which other systems or applications will be impacted?



**On Demand Insurance
CIO**

WebSphere Studio
Asset Analyzer can
help you



IBM

WebSphere Studio Asset Analyzer (WSAA)

- Powerful tool for Discovery and Impact Analysis
 - ▶ Search for application assets
 - ▶ Understand asset structure and relations
 - ▶ Perform data flow and impact analysis
 - ▶ Extract code for re-use
- Supports both mainframe and distributed assets
- Results are displayed in easy-to-read graphical windows
 - ▶ Web Interface available

WSAA Can Discover and Take Inventory of a Wide Variety of Mainframe Assets

- Source code: COBOL (including copybooks), PL/I, assembler, and JCL
 - ▶ Source code can be in a partitioned data set (PDS or PDSE) or controlled by a source control management (SCM) product.
- CICS online regions and transactions
- IMS subsystems and transactions
- SQL statements (DCLCURSOR, DCLTABLE, DELETE, FETCH, INSERT, SELECT, UPDATE)
- DB2 catalogs (columns, stored procedures, systems, tables, and views)
- MQ calls and queues

What Is the Impact of a Proposed Change? WSAA Provides Graphical Results

Context : [Home](#) [Impact analysis summary](#) [Impact analysis details](#)

Impact analysis details: Impact analysis results Actions

| Details | |
|-------------------------|--|
| Impact analysis: | QAD01:MASTER-STK-PART-NO |
| Description: | GENERATED for Program QAD01, Data element MASTER-STK-PART-NO |
| Type of asset analyzed: | Impact Analysis - Data element |
| Program/Element: | QAD01 /MASTER-STK-PART-NO |
| Created/last updated: | 3/22/05 4:51 PM by ASILBER / 3/22/05 4:53 PM by ASILBER |

Overview [Summary](#) [Details](#)

The following impact analysis diagram shows a subset of assets that this proposed code change directly and indirectly affects.

```

graph TD
    subgraph Direct_Impacts [Direct Impacts]
        D1[0 CICS transactions  
0 IMS transactions]
        D2[6 Batch jobs]
        D3[Starting with 1 data elements  
in 1 programs  
17 Data elements  
0 Entry points  
1 Other impacted programs]
    end
    subgraph Indirect_Impacts [Indirect Impacts]
        I1[2 CICS transactions  
0 IMS transactions]
        I2[1 Batch jobs]
        I3[29 Data elements  
2 Programs]
    end
    D1 --> D3
    D2 --> D3
    I1 --> I3
    I2 --> I3
    D3 <--> DS[(4 Data sets  
6 Data stores  
0 IMS segments  
0 DB2 tables)]
    DS <--> I3
  
```

The diagram illustrates the impact of a proposed code change on a central data store. It is divided into Direct Impacts and Indirect Impacts.

Direct Impacts:

- 0 CICS transactions
- 0 IMS transactions
- 6 Batch jobs
- Starting with 1 data elements in 1 programs
- 17 Data elements
- 0 Entry points
- 1 Other impacted programs

Indirect Impacts:

- 2 CICS transactions
- 0 IMS transactions
- 1 Batch jobs
- 29 Data elements
- 2 Programs

Central Data Store:

- 4 Data sets
- 6 Data stores
- 0 IMS segments
- 0 DB2 tables

Component Reuse through Code Extraction

- Use WSAA to identify a segment of code within a COBOL program that you want to reuse in an SOA, then select that code for extraction.
- WSAA will generate a program, and necessary data structures, and if possible a complement program to invoke the newly generated program.

The screenshot displays the WSAA Code Extraction tool interface, divided into two main panels: "Code extraction" and "Code extraction results".

Code extraction panel: This panel shows a COBOL source code snippet from the `ACCTINDX` source. The code is as follows:

```
62 | PROCEDURE DIVISION. 68000000
63 | OPEN INPUT ACCT-FILE 70000000
64 | OPEN OUTPUT ACIX-SAM. 72000000
65 | READ-MASTER. 74000000
66 | READ ACCT-FILE NEXT INTO ACCTREC 76000000
67 | AT END GO TO END-MASTER. 78000000
68 | MOVE CORRESPONDING ACCTREC TO ACIXREC. 80000000
69 | WRITE ACIXREC. 82000000
70 | GO TO READ-MASTER. 85000000
71 | END-MASTER. 88000000
72 | CLOSE ACCT-FILE. 91000000
73 | CLOSE ACIX-SAM. 94000000
74 | STOP RUN. 97000000
```

Instructions below the code state: "Select a line, range of lines (by holding the **Shift** key or dragging the mouse), or multiple ranges of lines (by using the **Ctrl** key) from the following `ACCTINDX` source." At the bottom of this panel are "Process" and "Close" buttons.

Code extraction results panel: This panel displays the results of the extraction. It includes a "Copy to clipboard" button and two tabs: "Extracted code" (selected) and "Compliment code". The extracted code is shown in a scrollable area:

```
1. IDENTIFICATION DIVISION.
2. PROGRAM-ID. EXTRACT.
3. /
4. ENVIRONMENT DIVISION.
5. /
6. DATA DIVISION.
7. WORKING-STORAGE SECTION.
8. LINKAGE SECTION.
9. ***
10. *** Input Declaration
11. ***
12. FD ACCT-FILE 46000000
13. LABEL RECORDS ARE STANDARD. 48000000
14. ***
15. *** Input/Output Declaration
16. ***
17. 01 ACIXREC. COPY ACIXREC. 62000000
18. ***
19. *** Input/Output Declaration
20. ***
21. 01 ACCTREC. COPY ACCTREC. 66000000
22. /
23. PROCEDURE DIVISION USING
24. ACCT-FILE
25. ACCTREC
26. ACIXREC
```

A "Close" button is located at the bottom of this panel.

Business Problem Solved

I am glad to see you have reduced our programming costs and our application backlog



**On Demand Insurance
CEO**

Increased productivity using WDz and WSAA made this possible. I didn't need to hire more System z development skills.



**On Demand Insurance
CIO**

Summary

WDz boosts developer productivity for both green screen and modern application development. WSAA brings a new level of capability to discovering and understanding your existing code.



IBM

