IBM

IBM Problem Determination Tools for z/OS

- Fault Analyzer demonstrations
- Debugging demonstrations
- Application Performance Analyzer demonstrations

Rufus Credle
Vikram Manchala
Dana Shelton
Kenichi Yoshimura

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM Problem Determination Tools for z/OS

January 2013

SG24-7918-02

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

Third Edition (January 2013)
This edition applies to File Manager Version 12.1, Fault Analyzer Version 12.1, Application Performance Analyzer for z/OS Version 12.1, Debug Tool Version 12.1, Rational Developer for System z Version 8.5, CICS Transaction Server Version 4.2, IBM® Workload Simulator for z/OS® 1.1.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

BladeCenter®
CICS Explorer®
CICS®
DB2 Universal Database™
DB2®
IBM®
IMS™
Language Environment®
MQSeries®
MVS™
NetView®
Optim™
OS/390®
Rational®
Redbooks®
Redpapers™
Redbooks (logo) ®
System x®
System z®
Tivoli®
VTAM®
WebSphere®
z/OS®

The following terms are trademarks of other companies:

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

IBM® Problem Determination (PD) Tools consists of a core group of IBM products that are designed to work with compilers and run times to provide a start-to-finish development solution for the IT professional. This IBM Redbooks® publication provides you with an introduction to the tools, guidance for program preparation to use with them, an overview of their integration, and several scenarios for their use.

If an abend occurs during testing, Fault Analyzer enables the programmer to quickly and easily pinpoint the abending location and optionally, the failing line of code. Many times, this information is all the programmer requires to correct the problem. However, it might be necessary to delve a little deeper into the code to figure out the problem. Debug Tool allows the programmer to step through the code at whatever level is required to determine where the error was introduced or encountered.

After the code or data is corrected, the same process is followed again until no errors are encountered. However, volume testing or testing with multiple terminals is sometimes required to ensure real-world reliability. Workload Simulator can be used to perform this type of testing.

After all of the tests are completed, running the application by using Application Performance Analyzer can ensure that no performance bottlenecks are encountered. It also provides a baseline to ensure that future enhancements do not introduce new performance degradation into the application.

In addition to the 3270 interfaces which continue to be enhanced, these tools can be started from the following eclipse-based workstation platforms:

- The PD Tools Studio
- Rational® Developer for z
- The CICS® Explorer® or the IMS™ Explorer

These options allow customers to easily address the needs of dynamic users, application developers, or system programmers.

This publication is intended for z/OS® application developers and system programmers.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Durham Center.

Rufus Credle is a Certified Consulting IT Specialist and Relationship Manager at the ITSO, Raleigh Center. He conducts residencies and develops IBM Redbooks and Redpapers™ about various topics, including network operating systems, high availability, IBM industry technology, IBM System x®, and IBM BladeCenter®. Rufus has worked for IBM for 32 years in various positions, including assignments in administration and asset management, systems engineering, sales and marketing, and IT services. He has a Bachelor of Science (BS) degree in Business Management from Saint Augustine's College.

Vikram Manchala is a senior developer at IBM Silicon Valley Labs working on IBM Problem Determination Tools. He has been at the forefront of IBM Problem Determination Tools GUIs and contributed to key features in the Debug Tool. He holds a Bachelors degree from National Institute of Technology, Warangal, India, and a Masters degree in Computer Science from Northern Illinois University.

Dana Shelton joined IBM more than 28 years ago. Marty has held positions as a programmer, Year 2000 Testing Architect, Project Manager, several first management positions, and a second line management position. For the past six years, Marty has been the Product Line Management for the IBM Problem Determination Tools. During Marty's entire tenure with IBM, he has worked out of the IBM Silicon Valley Lab (formerly Santa Teresa Lab), in San Jose, California.

Kenichi Yoshimura is a Development Manager within the IBM Australia Development Laboratory, and is responsible for the development and service of the File Manager for the IBM z/OS product. He has worked on a number of projects at IBM, including Fault Analyzer for z/OS and SCLM Developer Toolkit.

Thanks to the following people for their contributions to this project:

- IBM Silicon Valley Lab:
  - Francisco Anaya
  - Jesse Jackson
  - Yan Larsen
  - John Leake,
  - Eric Lin
  - Barbara Nardi
- IBM Deutschland GmbH:
  - Wilfried Van Hecke

Thanks to the authors of the previous editions of this book:

- Authors of the second edition, IBM Problem Determination Tools for z/OS, published in January 2012, were:
  - Chris Rayns
  - David Allen
  - Jeremy S Hamilton
  - Margaret Mortiz
  - Kenichi Yoshimura
- Authors of the first edition, IBM Problem Determination Tools for z/OS, published in September 2010, were:
  - Peter Henningsen
  - John Gallagher
  - Niel Kenyon
  - Speren Wilken

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

http://www.ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks
- Send your comments in an email to:

  redbooks@us.ibm.com
- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD 305/02/FF206
  3039 E. Cornwallis Rd
  Durham, NC 27709-2195

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:

  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm
- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Summary of changes

This section describes the technical changes that were made in this edition of the book and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes for *IBM Problem Determination Tools for z/OS*, SG24-7918-02, as created or updated on January 25, 2013.

January 2013, Third Edition

This revision reflects the addition, deletion, or modification of the following new and changed information:

New information

- Chapter 7
- Chapter 8
- Chapter 9

Changed information

- Chapter 1
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 6

1

Introduction to the IBM Problem Determination Tools

In this chapter, we introduce the following IBM tools that support application development on System z®:

- Application Performance Analyzer for IBM z/OS
- Debug Tool for z/OS and Debug Tool
- Fault Analyzer for z/OS
- File Manager for z/OS
- Workload Simulator
- IBM CICS Explorer and IBM IMS Explorer
- IBM Rational Developer for System z

These tools provide you with a comprehensive suite of products that can help improve application delivery and enhance production availability and performance.

1.1 Introduction

At the turn of the 21st century, IBM entered the problem resolution tool space for z/OS for the following reasons:

- To provide IBM customers with an alternative to excessively priced tooling for z/OS that were offered by independent software vendors (ISVs)
- To provide superior tooling in the problem resolution space

With more than 12 years in the market, we are confident to say that we accomplished both of these goals.

In this publication, we describe the following core IBM Problem Determination (PD) Tools (as shown in Figure 1-1 on page 3):

- IBM Application Performance Analyzer for z/OS
- IBM Debug Tool for z/OS
- IBM Fault Analyzer for z/OS
- IBM File Manager for z/OS
- IBM Workload Simulator

We consider their use when used with the PD Tools from a PD Tools Studio perspective, from a systems programming perspective with CICS Explorer and the IMS Explorer, and from a robust application development perspective with the Rational Developer for System z. These tools, together with the different GUI alternatives, provide a comprehensive set of problem determination tools to help improve application delivery throughout the application lifecycle.

Problem Determination Tools Version 12: Five Major Products

Key PD tools and functions:

- File Manager for z/OS Data management tool that supports key file structures, such as VSAM, DB2, CICS, and IMS.

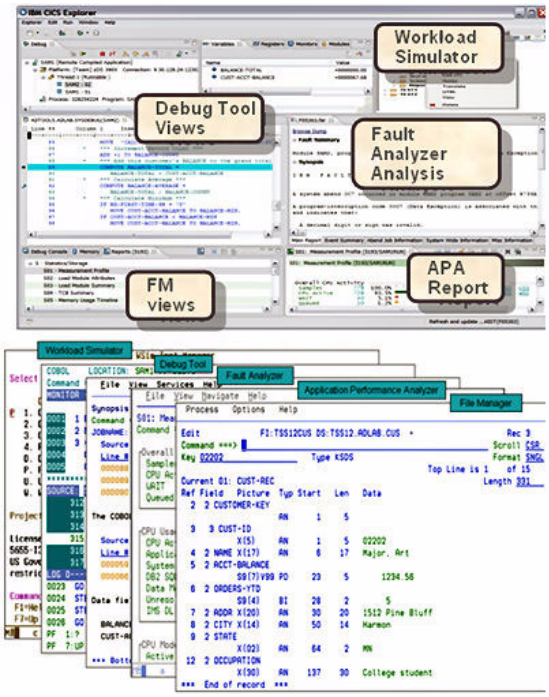
  New: DB2 and CICS Eclipse GUI Plug-Ins
- Debug Tool for z/OS Source code debugging to improve development productivity.

  New: Smooth debugging mixed apps JAVA/COBOL/PLI
- Fault Analyzer for z/OS helps you rapidly pinpoint the cause of failed applications (program abend)

  New: Java JVM Improvements
- Application Performance Analyzer for z/OS monitors performance at the application level.

  New: Expanded Java Support (JVM, WebSphere Application Server, CICS, IMS)
- Workload Simulator for z/OS and OS/390® performs regression and load testing of interactive z/OS applications.

  New: Initial Eclipse GUI Plug-Ins



Other related tools:

- Newly released Problem Determination Tools Studio GUI Offering
- Common components via CICS Explorer or IMS Explorer, including common login facilities and much.
- Rational Developer for System z: PD Tools GUI front-end plugs into Rational Developer for System z for a complete developer workbench for Cobol and Java.
- HourGlass: Date time manipulation
- ISPF Productivity Tool: Less key strokes for ISPF
- Optim™ TDM: Build of relational test data, subsets of production data, and scrambling

Figure 1-1 Problem Determination Tool V12

These tools help increase productivity and IT effectiveness across debugging, application abend analysis, data management, application-performance analysis, and the simulation of thousands of applications and users that drive your environment. IBM Problem Determination Tools do much more than support traditional applications. The tools also feature capabilities that you can use to build service-oriented applications. The tools also support and make the most of the latest subsystem levels and languages.

The IBM Problem Determination Tools provide support for all of the major IBM subsystems. Day one support is available for any new IBM z/OS subsystem release.

As businesses and the critical IT systems that support them grow increasingly complex, application developers are constantly struggling to meet the demands that are placed upon them. The use of service-oriented architecture (SOA) is widely accepted in IT because it fulfills the promise of helping to address these demands. The ability to reuse existing assets is the cornerstone of SOA. This possibility is significant because reusing assets can be less expensive than rewriting them. With the vast amount of existing applications that is running on the IBM System z platform, it makes sense that z/OS applications can be an important part of SOA.

1.2 IBM Application Performance Analyzer for z/OS

Application Performance Analyzer for z/OS helps programmers and systems personnel to identify constraints and improve the performance of their applications. It is a non-intrusive performance analyzer that helps you during the design, development, production, and maintenance cycles.

The key function of the product is to measure and report on how system resources are used by applications that are running in a z/OS address space, such as Time Sharing Option (TSO) and batch, and online subsystems. Examples of these subsystems include IBM IMS, Customer Information Control System (CICS), IBM WebSphere® Application Server, and IBM DB2® stored procedures.

You can monitor applications in test and production, and in multiple source languages, including Assembler, C/C++, COBOL, PL/I, and Java. Optimized code support for COBOL and PL/I is provided so you can monitor production applications.

The use of Application Performance Analyzer helps you maximize the performance of your existing hardware resources and improve the performance of your applications and subsystems. You can evaluate applications in the development and system test phases. You also can review the effect of increased data volume or changes in business requirements on performance, and generate historical data and reports to analyze performance trends and evaluate program changes.

Running in a separate address space, Application Performance Analyzer non-intrusively collects resource utilization, wait time, and statistical samples from the address space that is being monitored. This data is then analyzed and documented. The reports that are generated help you to identify the key performance bottlenecks that require examination and resolution. This data is available online and in printed reports that you can choose to create as PDF files for viewing on a workstation.

1.2.1 Application Performance Analyzer for z/OS V12.1

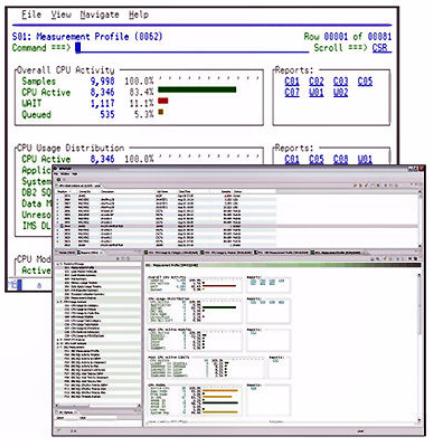
Pinpoint enterprise application bottlenecks

IBM Application Performance Analyzer Version 12

- IBM Application Performance Analyzer helps maximize the performance of your applications and improve the response time of your online transactions and batch turnaround.
- Identify constraints and improve the entire application's performance no matter where the problem is (CICS/IMS/DB2/MQ/COBOL/PLI/ASM/JAVA).
- Proven 3270-based interface and free graphical user interface.

New features:

- Enhanced support for WebSphere and Java.
- Sampling enhancements to improve the accuracy of reports.
- Support for CICS TS V4.2, IMS V12, and Java V6.0.1.
- Functional, usability, and security enhancements to the GUI plug-in that include improved integration with PD Tools Studio, CICS Explorer, IMS Explorer, and the other Problem Determination Tools family of plug-ins.



Monitor and optimize performance at the application level

Figure 1-2 IBM Application Performance Analyzer V12

When integrated with Application Performance Analyzer for z/OS (see Figure 1-2) and CICS Explorer, the IBM Application Performance Analyzer Plug-in for Eclipse encompasses the Observation Request and Reporting functions. The plug-in GUI can be used for submitting new observation requests and for navigating the performance analysis reports that are generated from observation requests. The plug-in GUI can display and provide functions to multiple components of Application Performance Analyzer simultaneously.

1.2.2 Subsystem support

In this section, we describe the following subsystem support topics:

- CICS

  Support for IBM CICS Transaction Server enables you to monitor specific CICS transactions by using wildcard transaction prefixes or by termID selection. With this support, you can trace transactions during critical situations, rather than waiting to review data that is collected periodically. Java application code that is running in the Java 2 Platform, Standard Edition (J2SE) environment under CICS Transaction Server is also supported. Multiple CICS regions can be measured simultaneously, and transaction data from these regions are merged to produce a set of CICS reports that show multi-region activity.

- DB2

  Support for DB2 delivers relevant information for performance analysis and tuning. This information includes SQL statements and processor usage by SQL statements and for IBM DB2 stored procedures that are written in a traditional language or in Java.

- IMS

  Support for IMS applications means that you can have IMS application performance data-on-call time and service-call time for Data Language Interface (DL/I). You can also choose to trace all IMS calls. A specific IMS transaction in a single IMS subsystem can be measured when it can run in multiple MPP regions.

- WebSphere MQ

  WebSphere MQ support provides information about CPU usage by queue, request, and transaction. Application Performance Analyzer also provides service time by queue, request, transaction, and wait time for the same categories.

- ADABAS/Natural

  Support is provided for sampling and reporting on ADABAS calls, if ADABAS=YES is set in the CONFIG SAMPLE statement. When a sampling request is created, an ADABAS extractor is presented if this configuration change is complete.

1.2.3 Java support

When enabled, the Java data extractor collects Java call stack information for each Java application thread. The call stack information identifies the methods in the call chain. Information about each method includes the package (if any), class, method, signature (parameter types and return type), and the source line number that is being executed (if available).

The generated reports include a summary, CPU usage, service times, and wait times.

HFS data is also collected and reported, if the Java data extractor is selected.

1.3 IBM Debug Tool for z/OS

IBM Debug Tool for z/OS provides debugging capability for applications that are running in various environments, such as IBM CICS, IBM IMS, IBM DB2 stored procedures, and IBM UNIX System Services. Debug Tool also includes features to help you identify old OS/VS and VS COBOL II source code and to upgrade the code to IBM Enterprise COBOL. These features meet the requirements of IBM DB2 Universal Database™ for z/OS, Versions 8 and 9, and IBM CICS Transaction Server for z/OS, Versions 3 and 4.

To effectively build and service applications, you require robust, easy-to-use tools to compile, test, and debug them. IBM Debug Tool for z/OS software provides a complete solution that can help you to reduce application development cycle times.

1.3.1 IBM Debug Tool for z/OS highlights

In this section, we describe the following highlights of the IBM Debug Tool for z/OS, as shown in Figure 1-3 on page 8:

- The IBM Debug Tool Plug-in for Eclipse, when integrated with Debug Tool for z/OS and CICS Explorer, debugs z/OS-based load modules that are running on z/OS. These load modules include modules that run in a subsystem, such as CICS, DB2, or IMS. You access the debugging capability that is similar to the capability that is accessed by the host full-screen interface by using the plug-in GUI interface. With this access, you can perform the following tasks:
  - Set and clear breakpoints at a specific line
  - Set and clear breakpoints for an error or warning-level error that is based on IBM Language Environment® severities
  - Run to a breakpoint
  - Step into a procedure
  - Step over a procedure
  - View variable values and change them as you step through the code
  - View variable values in the context of a larger area of storage
  - View the call stack

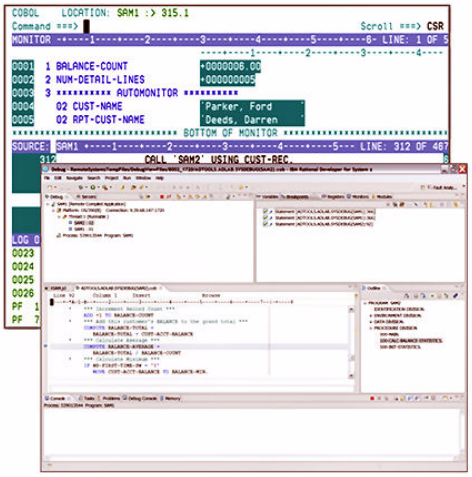
Debugging enterprise applications

IBM Debug Tool Version 12 provides the following features:

- Increase debugging efficiencies and reduces application development cycle times
- Program testing and analysis aid that helps you examine, monitor, and control running application programs on z/OS (CICS/DB2/IMS/ COBOL/PLI/ASM, C/C++/JAVA w Toolkit)
- Proven 3270-based interface and free graphical user interface

New features:

- Smooth debugging mixed apps JAVA/COBOL/PLI
- Enhanced performance for automonitor
- Support for CICS TS V4.2, IMS V12
- Functional, usability, and security enhancements to the GUI plug-in that include improved integration with PD Tools Studio, CICS Explorer, IMS Explorer, and the other Problem Determination Tools family of plug-ins
- A range of other enhancements to address key customer requirements.



Source code debugging to improve development productivity

Figure 1-3 IBM Debug Version 12

1.3.2 Support for IBM subsystems

Debug Tool works with individual load modules, independent of the subsystem in which they are running. Thus, Debug Tool provides functionality that can cross subsystems and languages on your z/OS system.

1.4 IBM Fault Analyzer for z/OS

Fault Analyzer for z/OS provides the information that you require to determine the cause, and assist with the resolution of, application and subsystem failures. You can use this tool to assist with composite-application abend analysis. The tool helps you repair failures quickly by gathering information about an application and its environment at the time of failure.

1.4.1 IBM Fault Analyzer for z/OS V12.1 highlights

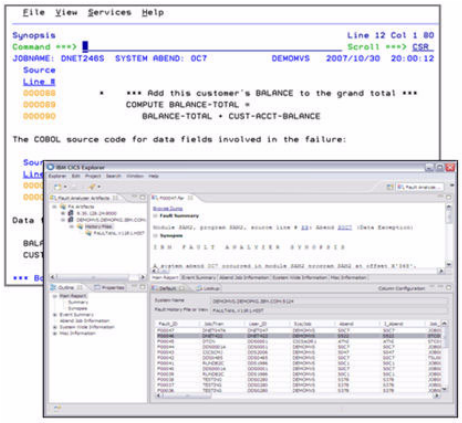
The IBM Fault Analyzer Plug-in for Eclipse, when integrated with Fault Analyzer for z/OS and CICS Explorer, provides access to problem reports for diagnosing mainframe application errors and abends, as shown in Figure 1-4.

Pinpoint the cause of enterprise application failures IBM Fault Analyzer Version 12

- IBM Fault Analyzer improves developer productivity and decreases deployment costs by helping to analyze and correct application failures quickly (CICS/DB2/IMS/MQ/COBOL/PLI/ASM/C/C++/JAVA).
- Develop and test new and existing applications more productively, which reduces costs.
- Proven 3270-based interface and free GUI.

New features:

- Enhanced Java support.
- Support for CICS TS V4.2.
- Functional, usability, and security enhancements to the GUI plug-in that include improved integration with PD Tools Studio. CICS Explorer, IMS Explorer, and the other Problem Determination Tools family of plug-ins.
- A range of other enhancements to address key customer requirements.



Helps to identify the cause, analyze the failure, and fix the problem

Figure 1-4 IBM Fault Analyzer

- IBM Fault Analyzer for z/OS includes the following key features:
  - An interface to manage views and multiple fault history files
  - The ability to browse fault entries that were created during real-time analysis of abending programs
  - A browser for browsing the dump storage that is associated with a fault entry
  - A source listing of abending programs that use side files
- IBM Fault Analyzer for z/OS includes the following Java support enhancements:
  - Integration of Java stack trace information into the Fault Analyzer event list
  - Java source support in which Java source is included in the abending JAR file

When an application abend occurs, Fault Analyzer captures and analyzes real-time information about the application and its environment. An analysis report that details the cause of the failure is then generated. The report describes the failure in terms of the application code, so you no longer lose time by reviewing cumbersome, low-level system error messages. By using Fault Analyzer, you can choose a report format to locate the information easily.

Each application abend is recorded by Fault Analyzer in a fault-history file by job name, failure code, and other details. The analysis report and storage pages that are referenced during the analysis also are included. This information can be later retrieved to reanalyze the failure.

Through the inclusion of information that is taken from selected manuals, Fault Analyzer can extract message and failure-code descriptions and insert them into the analysis report where applicable. You can also provide your own descriptions for messages.

You also can write your own user exits. For example, you can write a user exit to access compiler listings that are compressed or available only through a proprietary access method.

Integration through a GUI interface allows application developers to work with fault entries directly from their development environment.

1.4.2 Support for IBM subsystems

In this section, we describe the following topics in support of IBM subsystems:

- CICS

  Fault Analyzer functions are available from within the CICS transaction-based environment in a manner that is consistent with the ISPF usage. You can review and analyze fault entries in history files without the need for a TSO logon.

  Fault Analyzer provides general information about a CICS region and details of control blocks, transaction storage, the last screen buffer, the trace table, and an explanation of the CICS abend code.

  The ability of Fault Analyzer to detect and analyze dump formatting and storage violations is helpful for system programmers who debug CICS system problems. Options are provided to speed the processing of duplicate dumps and to skip analysis of repeated abends.

- DB2

  Details of the last SQL activity are provided. Plan and package information also is included.

- IMS

  Fault Analyzer provides general information about an IMS region, details of the last DL/I call parameter list, information for all PCBs in the order of their relative PCB number and, if available, JCB call trace information. IMS accounting information also is provided.

  You can eliminate the overhead of duplicate IMS faults that occur across all IBM MVS™ images in a sysplex.

- WebSphere MQ

  API information and return-code diagnostic procedures are provided to help with problem determination of WebSphere MQ applications.

1.4.3 Java support

When Fault Analyzer is started from within a Java application, abending traditional programs (COBOL, PL/I) can be analyzed.

A feature unique to the interactive component of Fault Analyzer is the ability to analyze information that is related to Java. The Java execution might be under WebSphere, CICS, or UNIX System Services on MVS. Typically, the environment is Java calling legacy programs. How to perform the following tasks is explained in the User's Guide and Reference manual:

- Set options for Java analysis
- Select a Java dump data set for analysis
- Display the resulting Java information in the interactive report
- Create a history file entry for the analyzed dump data set

1.4.4 Analysis options

Fault Analyzer provides three modes to help you better track and analyze application and subsystem failure information. Reports that are generated can be viewed through the SDSF or through the Fault Analyzer ISPF, CICS Explorer, or Rational Developer for System z interface.

Real-time analysis

When an application failure occurs, the Fault Analyzer exit starts real-time analysis. After failure processing, you can view the analysis report in your TSO session or in a printed report.

Batch reanalysis

Batch reanalysis generates a new analysis report. This report is based on the dump and information that is gathered in real time, but with potentially different options that are specified, or with compiler listings or side files that are made available. You can submit a Fault Analyzer batch-mode job by using the Fault Analyzer ISPF or your own job control language.

Interactive reanalysis

Interactive reanalysis runs under ISPF and CICS, which you use browse through a formatted, structured view of a fully detailed reanalysis. By using this Fault Analyzer mode, you can view working storage and control blocks at the time the dump was written. The interface has many point-and-click fields for easy navigation through the interactive reports.

1.5 IBM File Manager for z/OS

File Manager for z/OS offers comprehensive tools for working with z/OS data sets, CICS resources, DB2 data, and IMS data.

Extending the standard browse, edit, copy, and print utilities of ISPF, File Manager features tools that support personnel find useful. It also delivers enhancements that are designed to address the requirements of application developers who are working with files that contain structured data. Also, because the basic features of the File Manager editor and the ISPF/PDF editor are deliberately almost identical, you can take advantage of extra features without having to relearn fundamental skills.

File Manager includes the following components:

- A base component to manage z/OS data sets, such as queued sequential access method (QSAM), VSAM, partitioned data sets (PDS), and IBM z/OS UNIX System Services hierarchical file system (HFS) files
- A DB2 component to manage DB2 data
- An IMS component to manage IMS data
- A CICS component to manage data in CICS VSAM file resources, transient data queues, and temporary data queues

Important: You install only the components that are relevant to your environment.

1.5.1 IBM File Manager for z/OS V11.1 highlights

With File Manager V12.1 (see Figure 1-5), many more capabilities were added. The Problem Determination Tools Studio and the IMS Explorer for invocation of mainframe FM product was added to the capabilities that are provided by CICS Explorer and Rational Developer for System z.

In addition, the Eclipse-based File Manager DB2 support and CICS support was added in Version 12. IMS Version 11.1 support also was added to the mainframe FM product. Many other functions and capabilities also were added to File Manager V12.1.

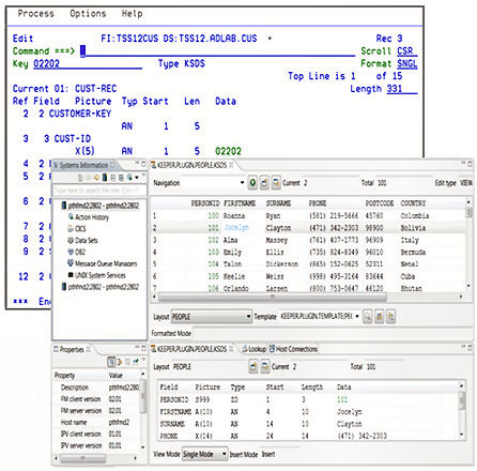
Manage various enterprise data file structures

IBM File Manager Version 12 includes the following features:

- Manage production, test, and development data across multiple formats and storage media.
- Create, edit, copy, browse, extract, print, and compare enterprise data (VSAM/DB2/IMS, CICS/MQ)
- Proven 3270-based interface and free graphical user interface.

New features:

- GUI plug-in support for File Manager/DB2
- GUI plug-in support for File Manager/CICS
- Support for IMS V12
- Formatted KSDS key
- Functional, usability, and security enhancements to the GUI plug-in that include improved integration with PD Tools Studio. CICS Explorer, IMS Explorer, and the other Problem Determination Tools family of plug-ins
- A range of other enhancements to address key customer requirements



Data management tool supporting key file structures like VSAM, DB2, CICS, and IMS

Figure 1-5 IBM File Manager

1.5.2 File Manager base component

The File Manager base component helps speed the application-development process by identifying the structure of your records and displaying each field in a human readable format, according to its data type.

This component supports VSAM, including Tivoli® Innovation Access Method files; QSAM, PDS, and WebSphere MQ queues; and z/OS UNIX System Services hierarchical file system (HFS) data sets, including support for double byte character set (DBCS) data in these data sets.

You can edit entire files (regardless of size) and use a template or copybook for formatting and record selection during an edit or browse session.

1.5.3 Support for IBM subsystems

In this section, we describe the following topics in support of IBM subsystems:

- CICS

  Users can manage or query data in CICS VSAM file resources, transient data queues, and temporary data queues. The File Manager for CICS feature allows access to CICS resources under a CICS transaction. The CICS resources that are supported for view, edit, and certain File Manager utilities (such as data create, copy, and compare) are VSAM files, temporary storage queues, and transient data queues.

  You can authorize users to view and change the status of supported local and remote CICS resources and perform File Manager base and File Manager IMS tasks from the File Manager CICS transaction.

  The same display and record selection capabilities are present in the CICS environment as are in the BASE product, which provides for quick easy access to CICS data. Users also can create full copies of open CICS files (or TS or TD queues) to another QSAM, VSAM, HFS, or PDS file.

  Most of the File Manager functionality that is related processing happens offline to the CICS task. Thus, File Manager runs little risk of adversely affecting other CICS users, even when an entire file is copied somewhere else, for example.

  File Manager for CICS maintains the same look and set of editor commands that you might be familiar with in an ISPF environment. It is not necessary to log out of your CICS environment to log on to your ISPF, if you want to use the File Manager BASE or IMS components.

- DB2

  Whether you are a DB2 DBA, application programmer, or want to retrieve information that is stored in a DB2 database, the File Manager DB2 component provides something for you.

  Included are database management abilities (such as creating and dropping objects, copying data within DB2, handling privileges, and import/export). There is an editor for DB2 tables and views (in read-only or update mode), which encompasses all of the common Insert, Delete, and Update functionality that is often required in a database application.

  Full tables, views, or the results of a limited SQL query that you customized are provided (including the ability to save the query you made) in a File Manager template.

  If you are writing SQL to be used in your applications, another handy use of File Manager is the ability to refine and test an SQL query by the prototyping and analysis feature, which includes an explanations feature.

  Getting data out of, or back into, DB2 to or from QSAM or VSAM files also is provided for by a powerful utility. DB2 data can be exported in the following formats:

  – The external format that is used by the DB2 UNLOAD utility.
  – The format that is used by the DB2 sample unloads program DSNTIAUL.
  – A File Manager export format that can be tailored with multiple options to control the handling of NULL values and options for various column data types.
  – A delimited text format, such as comma-separated value output.

  You also can generate batch JCL for the most commonly used DB2 utilities to save time.

- IMS

  With File Manager's IMS component, it is easy to access data that is stored in an IMS database. Although the database storage concepts are different from DB2, File Manager delivers a similar look when your data us viewed or edited. Also, creating and saving customized queries is as easy as using the other components.

  By using record structures that are defined in COBOL or PL/I copybooks (similar to the BASE component), the File Manager IMS component enables you to view or edit IMS segments that are displayed with formatting according to their individual field data types. You can find and change data across an entire database hierarchy or in specified segments and fields.

  Data displays for IMS always include the segment name to help you identify where you are in the database. This feature is useful when you are viewing a large result set or an entire hierarchy.

  Database navigation commands are available with flexible parameters and scoping to allow selective navigation or applicability, or a more generic approach, depending on your requirements.

  You can use flexible criteria to select IMS database segments that contain specific field values and extract the segments into a sequential data set for later use. You also can use the IMS component to load them into a different database.

- WebSphere MQ

  From within File Manager, you can access WebSphere MQ queues on the local z/OS system where File Manager is running. From the Primary Option Menu panel, select option 9, **WebSphere MQ**. File Manager then displays the WebSphere MQ Functions panel, from which you can perform the following tasks:

  – List managers and queues
  – View a WebSphere MQ queue
  – Edit a WebSphere MQ queue

1.5.4 Java support

File Manager can access an HFS file as a simulated QSAM/BSAM file. This access means that at the I/O level, the file is seen as a single-volume, physical, sequential data set that is on a direct access storage device.

1.5.5 SOA support

By using File Manager, you can generate XML data from files. A File Manager template that describes the data-record layouts is required. The XML tags are generated based on the field names from the template, and the XML content comes from the data. A number of options are provided for handling invalid and unprintable data.

File Manager provides a plug-in for integration with Rational Developer for System z, allowing all aspects of web-service (and traditional application) development to be undertaken from the same developer tool.

File Manager provides a scrambling algorithm that modifies data in a field and maintains its system data type. Scrambling is intended to de-identify (disguise) personal information in different ways for different data types. The goal of repeatable scrambling is that application relationships that are based on equality tests can be maintained, if wanted, even after the data is scrambled. A number of standard algorithms are provided to give the user complete control over the type of scrambling performed.

1.6 Workload Simulator

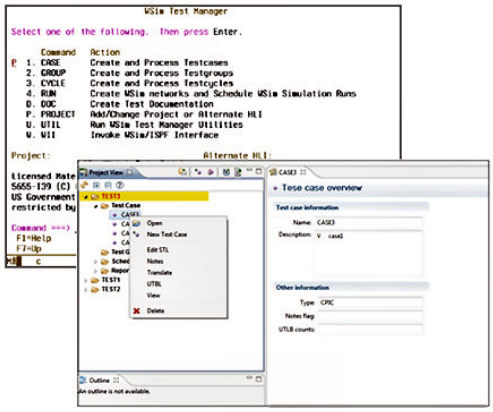
IBM Workload Simulator (as shown in Figure 1-6) simulates terminals and the associated messages. The user also can alter message loads during a run. It can be used to generate a large volume of messages to evaluate the reliability and approximate performance characteristics of a network under expected operating conditions. Anything that a real user can do at terminal, Workload Simulator can do faster, more reliably, and often at less cost.

Regression and load test interactive z/OS applications
IBM Workload Simulator

- IBM Workload Simulator helps eliminate the need for large amounts of terminal hardware and terminal operator time for testing.
- It is ideal for stress, performance, regression, function, and capacity planning tests.
- Proven 3270-based interface and includes a free GUI.

New feature:

- Initial release of free GUI provides access to the power of the host product.



Simulates a network of terminals and the associated messages

Figure 1-6 IBM Workload Simulator

IBM Workload Simulator includes the following benefits and features:

- Helps prepare your networks for peak transaction volumes.
- Enables testers to conduct reliable tests on stress, performance, and capacity and regression and function tests.
- Simulates different terminals, terminal features, and terminal operator actions.
- Provides support for enhanced TCP/IP support, SNA, and CPI-C (LU 6.2).
- Helps manage the test process.
- Offers several options for creating scripts to use in simulations.
- Provides screen images, data, and reports during simulation.
- Helps testers compute and analyze test results.

You can run Workload Simulator on MVS in batch mode as a procedure by using the Workload Simulator/ISPF Interface, or under TSO.

1.7 IBM Problem Determination Tools Studio

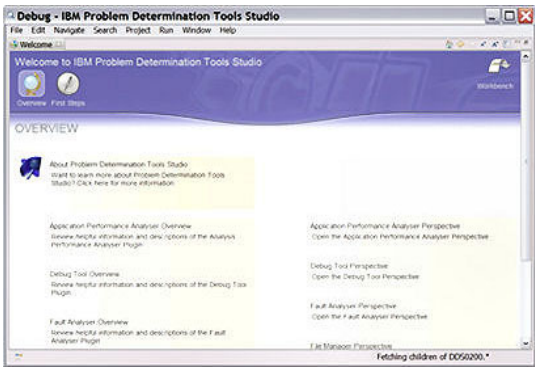
With Version 12 of the PD Tools, the Problem Determination Tools Studio (as shown in Figure 1-7 on page 17) has arrived. This self-contained, downloadable application was designed for customers who must use our unified Eclipse-based plug-ins in problem determination-specific environments. We are excited about being able to offer our customers an alternative interface to the traditional 3270, or green screen interface. The PD Tools Studio is available to any licensed customer of our mainframe Problem Determination Tools products at no additional charge. The following Eclipse interfaces are included in the downloadable application:

- Application Performance Analyzer
- Debug Tool
- Fault Analyzer
- File Manager
- Workload Simulator

In addition to the plug-ins, simple editing, submission capabilities, and Job Entry Subsystem (JES) access are available. A common server and common login capabilities are also provided to any sanctioned PD Tools Studio user.

Problem Determination Tools Studio –
Full-function, Eclipse-based GUI Environment

- **PD Tools Studio**
  - Includes five individual V12 refreshed PD Tools Plug-ins:
    - Debug Tool
    - Fault Analyzer
    - File Manager
    - Application Performance Analyzer
    - Workload Simulator
  - Improved and simplified connections management
  - Intuitive and secure access to z/OS resources
  - Easier installation and maintenance
  - Expandable to incorporate other plug-ins (for example, CICS Tools)



- **PD Tools Studio Includes:**
  - APA - used for submitting new observation requests and navigating performance analysis reports.
  - DT - helps debug load modules running on z/OS, including those run in CICS, DB2, and IMS.
  - FA - access to problem reports for diagnosing mainframe application errors and ABENDs.
  - FM - an interface to a set of tools for manipulating and browsing data stored in various z/OS resources including data sets, USS files, and MQ queues.
  - WS - helps manage the test process and automatically generate Workload Simulator scripts and networks.
  - z/OS Perspective - A perspective and views included to list, create, edit, and control z/OS datasets, PDS datasets, zFS files and paths and to submit jobs and view JES spool output.

Figure 1-7 Problem Determination Tools Studio

1.8 CICS Explorer and IMS Explorer

With Version 12 of the PD Tools, several updates and new capabilities are provided for systems-related programmers and developers.

The Problem Determination Tools plug-ins (as shown in Figure 1-8 on page 18) evolved and is used with the CICS Explorer, and CICS TS 3, 4, and the latest CICS TS 5.1. The addition of the ability for PD Tools plug-ins to be used with the new IMS Explorer is another great advancement.

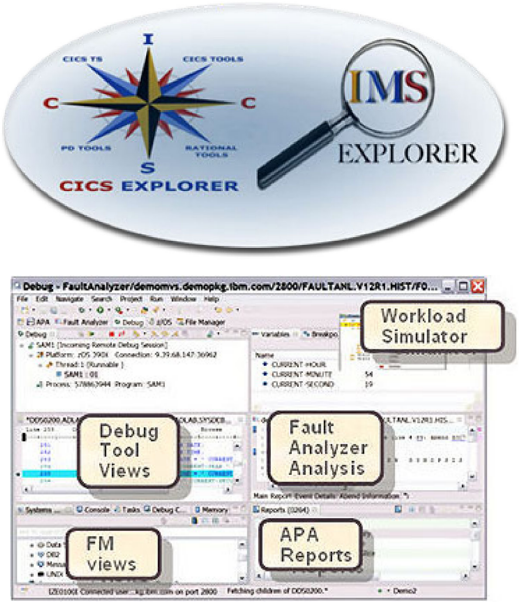
As with the PD Tools Studio, editing and submission capabilities, JES access, a common server, and common login capabilities are available to any sanctioned CICS or IMS user. The plug-ins also are available to any licensed holder of the PD Tools mainframe-equivalent Application Performance Analyzer (APA), Fault Analyzer (FA), File Manager (FM), Debug Tool (DT), or Workload Simulator (WSIM) products (as shown in Figure 1-8).

Premier System Programming or Stand Alone

Environment with IBM's Problem Determination Tools

Use PD Tools Plug-ins in conjunction with CICS Explorer or IMS Explorer

- CICS/IMS Explorer is now fully supported for users of CICS Transaction Server V3 and V4.
- CICS/IMS Explorer now provides an intuitive and secure way to view and edit z/OS datasets and zFS files, submit JCL, and view output and job logs. These capabilities are exploited by the plug-in for CICS Deployment Assistant.
- V12 PD Tool plug-ins have been refreshed, with the addition of Workload Simulator.
- z/OS Explorer V1.1 is also embedded in IMS Enterprise Suite Explorer for Development V2.1.2, giving the PD Tools plug-ins a consistent user experience.



- New functions with CICS Explorer and IMS Explorer:
  - Copying and pasting resource data to a spreadsheet.
  - Exporting a file or folder to a z/OS Unix file system.
  - Creating a new z/OS Unix file.
  - Creating a new data set.
  - Ability to create a new resource from an existing definition.
  - Ability to see the contents resource definition group.
  - Support for multiple selection job submissions from the z/OS Data Sets view.
  - Support for deleting or purging multiple files in the z/OS Explorer.
  - Ability to create a Job Submission project and JCL files, in the Project Explorer view in the resource perspective, while working offline.

Figure 1-8 CICS Explorer, IMS Explorer, and PD Tools Plug-ins

CICS Explorer, first introduced in 2009 as the new face of CICS, provides an integration point for CICS tooling with rich CICS views, data, and methods. CICS Explorer includes the following features:

- A common, intuitive, Eclipse-based environment for architects, developers, administrators, system programmers, and operators.
- Task-oriented views provide integrated access to a broad range of data and control capabilities.
- Powerful, context-sensitive resource editors.
- An integration point for CICS TS, CICS Tools, CICS Transaction Gateway, the IBM Problem Determination Tools, and Rational Tools.
- Extension by independent software vendors, system integrators, and customers by using the Java Software Development Kit.

The CICS Explorer code is downloadable for free from the CICS Explorer website:

http://www-01.ibm.com/software/htp/cics/explorer/

Important: This code is refreshed approximately every three months.

IMS Enterprise Suite Explorer for Development

Information Management System (IMS) Enterprise Suite Explorer for Development (IMS Explorer for Development) is the new face of IMS. It is a simple, no-charge, essential tool with which you complete common application development tasks with ease.

If you are an IMS application developer or database architect, you benefit from IMS Explorer's lightweight, powerful, extensible framework, which seamlessly integrates with your other Eclipse-based tools. With IMS Explorer, you can perform the following tasks:

- Integrate with the IMS catalog and simplify your application tasks.
- Quickly and easily model, display, and edit IMS database and program definitions.
- Use new structure and array types that are accessible because of IMS catalog support.
- Simplify segment mapping.
- Gain a relational view of IMS data.
- Securely and intuitively view and edit z/OS data sets and z/OS UNIX files, submit JCL, and view output and job logs.
- Build SQL statements to use with IMS data.
- Use rich GUI controls and context-sensitive help.

The IBM CICS Explorer code is downloadable for free from the CICS Explorer website:

http://www-01.ibm.com/software/htp/cics/explorer/

1.8.1 Support for the IBM Problem Determination Tools

The IBM Problem Determination Tools plug-ins, when used with CICS Explorer, provide easy access through a GUI on the workstation to the power of the following host IBM Problem Determination Tools:

- Application Performance Analyzer for z/OS
- Debug Tool for z/OS
- Fault Analyzer for z/OS
- File Manager for z/OS
- IBM Workload Simulator for z/OS

Plug-ins for each of these tools are available for download for free from this website:

http://www-01.ibm.com/software/awdtools/deployment/pdtplugins/

1.8.2 Support for CICS Tools

The following CICS Tools are supported through plug-ins to CICS Explorer:

- Configuration Manager
- Interdependency Analyzer
- Performance Analyzer
- IBM Session Manager

For more information about these plug-ins, see the CICS Tools website:

http://www-01.ibm.com/software/htp/cics/tools/

To download the plug-ins, see this website:

http://www-01.ibm.com/software/htp/cics/explorer/

1.9 Rational Developer for System z

Rational Developer for System z speeds the development of traditional mainframe, web, and composite applications. Built on an Eclipse platform, Rational Developer for System z lends itself readily to integration with IBM and non-IBM products, as shown in Figure 1-9.

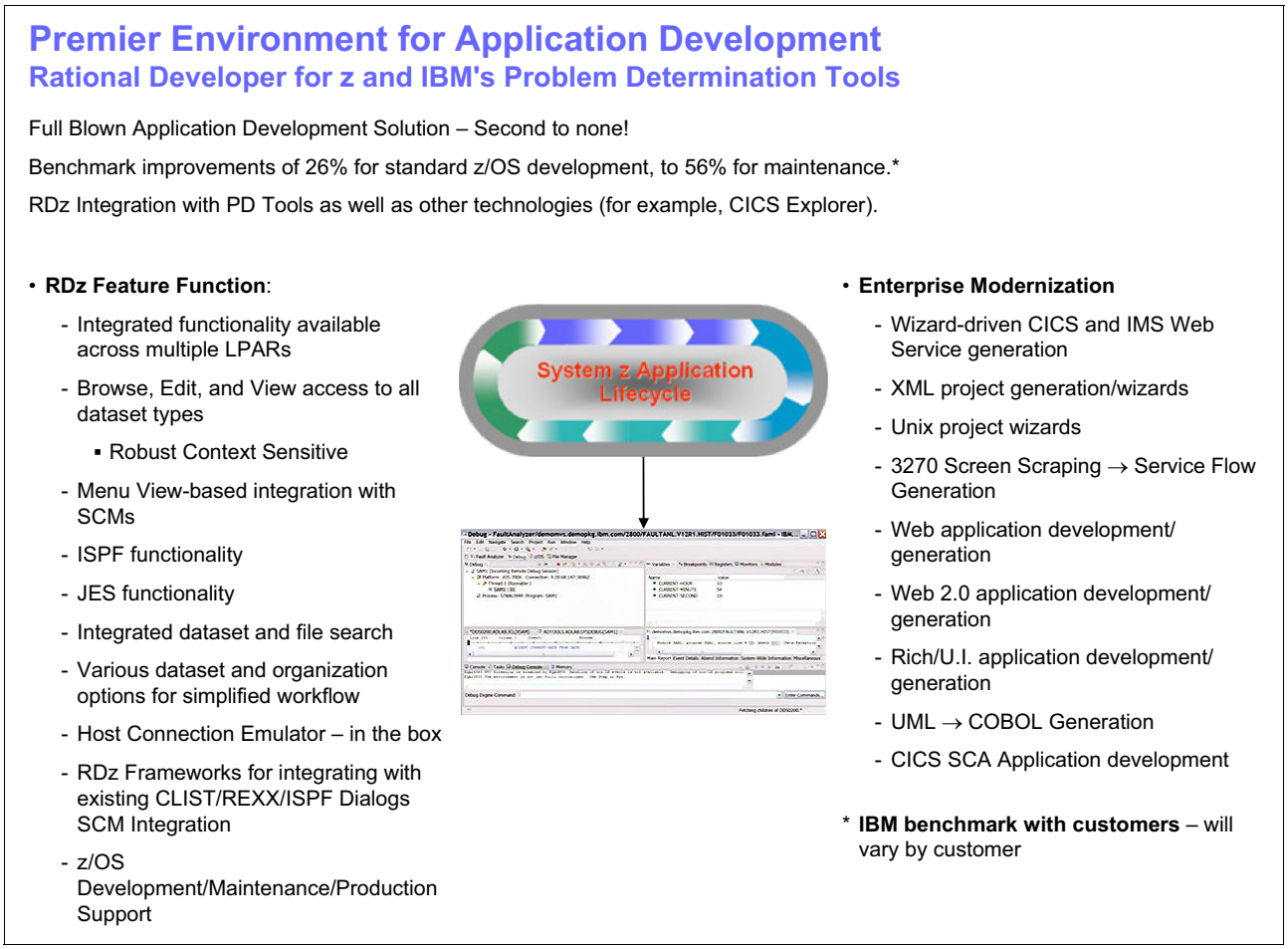


Figure 1-9 Premier Environment for Application Development

1.9.1 Support for IBM Problem Determination Tools

In this section, we describe the IBM Problem Determination Tools.

Debug Tool

COBOL and PL/I applications, EGL applications, and Java applications can be debugged through a common interface by using the local debugger. The remote debugger supports debugging of code that runs in the following z/OS environments:

- Batch
- TSO
- CICS
- IMS (IMS Database Manager and IMS Transaction Manager), with or without Batch Terminal Simulator (BTS)
- DB2 (including stored procedures)
- WebSphere Application Server

The debugging sessions are cooperative, meaning that the remote distributed debugger is on the workstation and interfaces with the IBM Debug Tool, which runs on the host with your application. The workstation interface communicates with the host z/OS products through TCP/IP.

Fault Analyzer

Fault Analyzer (FA) integration allows users to perform the following tasks:

- Browse FA abend reports
- View dump selections that are related to abends
- Annotate reports for future reference or to share comments with other users who browse the same reports

File Manager

File Manager (FM) integration enables access to perform the following tasks:

- VSAM KSDS files for browsing and updating
- Template-driven display of VSAM, PDS members, and sequential file data

1.9.2 Remote compile generation

With remote editing, compiling, and debugging, you can develop or enhance many types of applications, including CICS, batch, TSO, or IMS Transaction Manager applications. These applications can access many forms of data, such as DB2, VSAM, DL/I, and QSAM data. You can save time by editing, compiling, and debugging host applications remotely. When host files are edited, compiled, and debugged from the workstation, you work in a cooperative Windows and TCP/IP-based development environment, which avoids lengthy downloads and uploads unless explicitly wanted.

1.10 IBM Problem Determination Tools GUI options summary

IBM offers its customers alternative GUI interfaces and GUI technologies in addition to the traditional 3270 or green screen, which IBM continues to progress. For users who want to use the PD Tools in a unified Eclipse-based problem determination environment, the Problem Determination Tools Studio can address your needs as shown in Figure 1-10. Customers who must use the Problem Determination Tools in a full Eclipse-based application development environment can take advantage of the PD Tools plug-ins with Rational Developer for z. However, if you would like to use the PD Tools plug-ins in systems-related environment, you can use the tools with the CICS Explorer or IMS Explorer. All alternative interfaces give you easy access to the host systems PD Tools, such as Application Performance Analyzer, Debug Tool, Fault Analyzer, File Manager, and Workload Simulator through a GUI on the workstation.

Figure 1-10 Problem Determination Tools V12

When the plug-ins are used with PD Tools Studio environment, CICS Explorer, or IMS Explorer, no additional charge is required if the customer is licensed to one or more of the PD Tools mainframe products.

With the PD Tools Studio, Rational Developer for z, and the CICS/IMS Explorer interfaces, IBM is giving customers several alternatives to meet your unique enterprise needs.

1.11 IBM Problem Determination Tools in summary

IBM Problem Determination Tools includes the following applications:

- Application Performance Analyzer
- Debug Tool
- Fault Analyzer
- File Manager
- IBM Workload Simulator

These tools, along with CICS Explorer and Rational Developer for System z, are designed to help ease the burden of developing, testing, and supporting service-oriented and composite applications across complex IBM System z environments.

By helping to improve application delivery throughout the application lifecycle, these tools provide increased user productivity and IT effectiveness across source code debugging, application-abend analysis, data management, and application-performance management.

System z tools, including the Problem Determination Tools, CICS Tools, and application development tools, support the entire application lifecycle to help you build, integrate, test, and manage enterprise solutions. As a result, you can make the most of your System z platform investments and expedite your move to SOA. With these tools, you also can transform your applications and optimize your IT operations to achieve greater business flexibility, without affecting governance and compliance.

2

Introduction to the IBM Problem Determination Tools Plug-ins

In this section, we present an overview of the IBM Problem Determination Tools Plug-ins, which are available in the following formats:

- The IBM Problem Determination Tools Studio

  This format is a stand-alone application that contains all of the PD Tools plug-ins, which you can download and use without installing any other software.

- The IBM Problem Determination Tools SDK (P2)

  This format is a compressed file that contains all of the PD Tools plug-ins. You can install the plug-ins by using Eclipse P2 installer. However, you must select a shell Eclipse environment to install the plug-ins, such as IBM CICS Explorer and IBM IMS Explorer.

- The IBM Problem Determination Tools SDK (Installation Manager)

  This format is a compressed file that contains all of the PD Tools plug-ins. You can install the plug-ins by using IBM Installation Manager into Rational Developer for System z.

The choice of format depends on your circumstance. If you are interested in getting started with the PD Tools plug-in, IBM Problem Determination Tools Studio is the easiest option.

If you are interested in taking advantage of the system administrative functionality that is available in CICS Explorer or IMS Explorer, IBM Problem Determination Tools SDK (P2) is the best choice.

If you are interested in advanced application development, IBM Problem Determination Tools SDK (Installation Manager) is the best choice.

2.1 The IBM Problem Determination Tools Studio

The IBM Problem Determination Tools Studio is a self-contained application that contains all of the PD Tools plug-ins in a basic Eclipse 3.6.2 platform. If you are interested in using the PD Tools plug-ins capabilities as quickly as possible, this option is best for you. The following plug-ins are included in the studio:

- Application Performance Analyzer for z/OS plug-in
- Debug Tool for z/OS plug-in, including Debug Tool Startup Profile (DTSP) and Debug Tool for Console (DTCN) plug-ins
- Fault Analyzer for z/OS plug-in
- File Manager for z/OS plug-in
- Workload Simulator for z/OS plug-in

2.1.1 Installation instructions

Complete the following steps to install the IBM Problem Determination Tools Studio:

1. Download the `PD_Tools_V12xxxx.zip` file from this website:

   `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`

2. Extract the compressed file into a folder of your choice (you can delete the compressed file after the files are extracted).
3. In the folder, you see an executable file called `PD_Tools.exe`. Double-click the file to start the studio.
4. You see the splash window. Wait for the application to load, as shown in Figure 2-1.

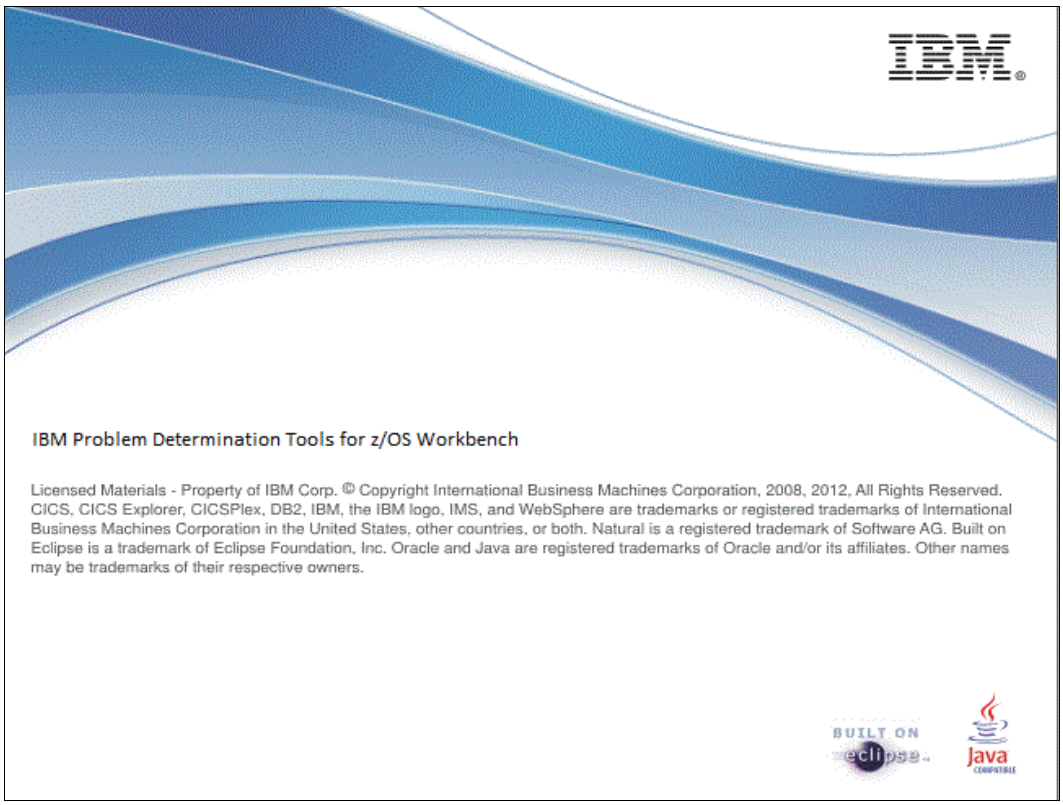


Figure 2-1 The IBM Problem Determination Tools Studio splash window

5. Specify the directory in which you want the studio to store projects and files, as shown in Figure 2-2.

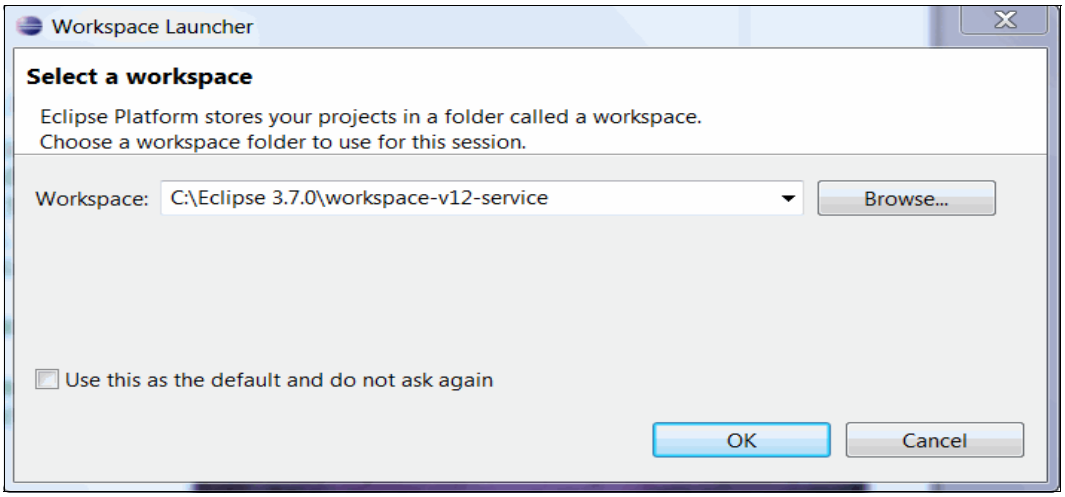


Figure 2-2 Workspace selection dialog

6. A welcome window provides an overview of the PD Tools plug-ins, as shown in Figure 2-3.

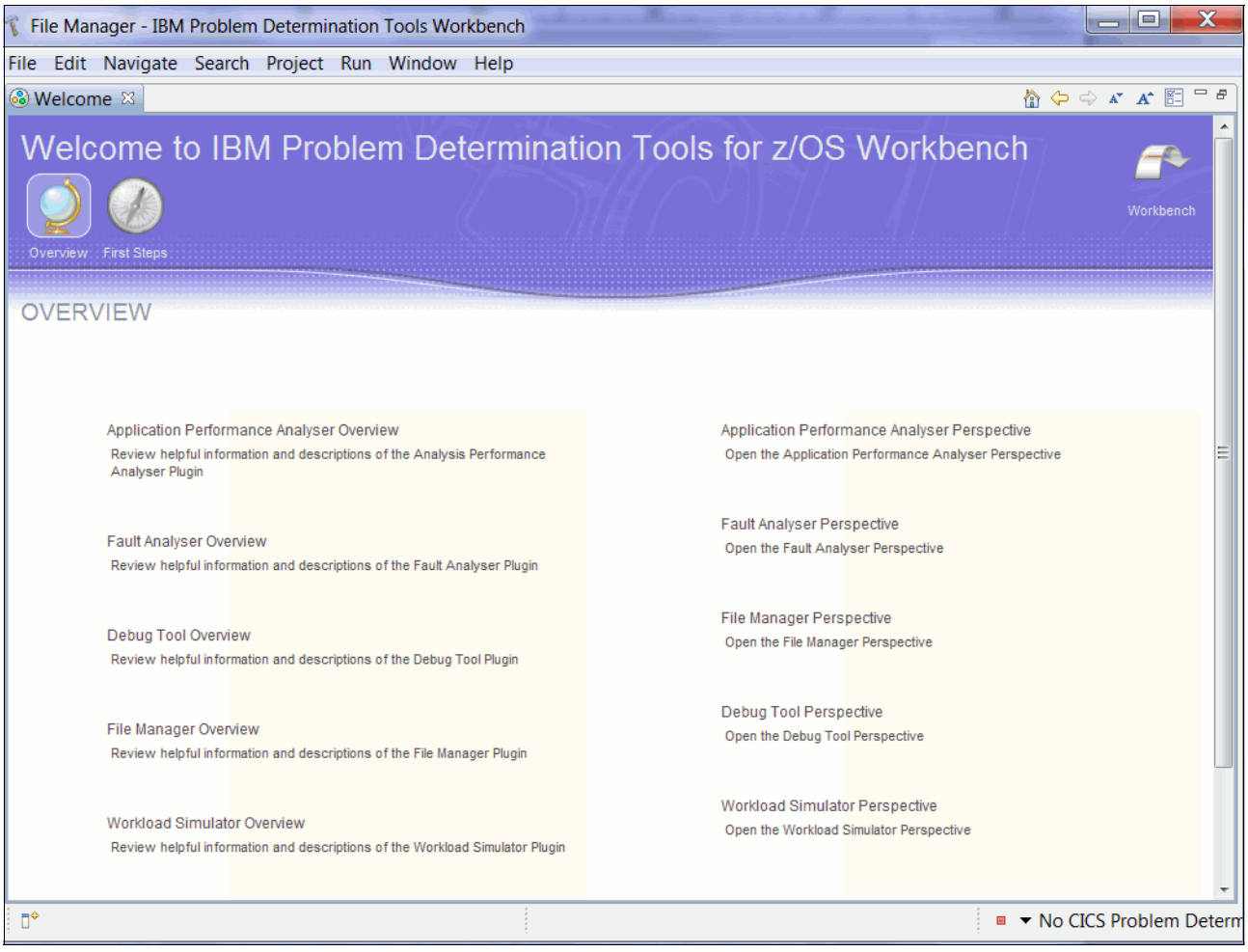


Figure 2-3 IBM Problem Determination Tools Studio Welcome window

The welcome window gives you a quick overview of the IBM Problem Determination Tools Studio and links to more information about individual product pages. From the individual product welcome windows, links are provided to access the corresponding online documentation and for opening their perspectives.

2.1.2 Updating the IBM Problem Determination Tools Studio

You have two options to use to update the IBM Problem Determination Tools Studio:

- IBM Problem Determination Tools SDK (P2)
- Eclipse update facility

These options are described next.

Update plug-ins by using the IBM Problem Determination Tools SDK (P2)

The IBM Problem Determination Tools Studio includes the Eclipse P2 installation feature, which you use to add and update plug-ins that are installed in the environment. You can update the PD Tools plug-ins that are installed in your studio environment by using the IBM Problem Determination Tools SDK (P2) which is described in 2.2, “IBM Problem Determination Tools SDK (P2)” on page 32.

Complete the following steps to update the plug-ins:

1. Download the `PD_Tools_SDK_V12xxxx.zip` file from this website:

   `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`
2. Select **Help** → **Install New Software...**.
3. Select **Add** to specify the compressed file that was downloaded in step 1.
4. Select the plug-ins to install/update, as shown in Figure 2-8 on page 34.
5. Click **Next** to review the installation information.
6. Click **Next** again to review the license agreement.
7. Click **Finish** to complete the installation/update.

Update plug-ins by using Eclipse update facility

The IBM Problem Determination Tools Studio is pre-configured with our update site. You can check for updates by completing the following the steps:

1. Select **Help** → **Check for Updates**.
2. If any updates are found, an option to update the plug-in is presented.

All known plug-in update sites are now connected through your studio environment, including the PD Tools plug-ins update site. After the updates are found, you can choose to update the plug-ins.

2.1.3 Extending the capability of the IBM Problem Determination Tools Studio

After you start to use the IBM Problem Determination Tools Studio, you might want to expand the capability of the studio by installing more Eclipse plug-ins. The studio is equipped with Eclipse P2 Installation and Update facility, with which you can expand these capabilities. In this section, we describe the steps that are used to add the CICS Explorer capabilities.

Complete the following steps to install the CICS Explorer capabilities:

1. Download a copy of CICS Explorer SDK from this website:

   http://www.ibm.com/software/htp/cics/explorer/

2. Select **CICS Explorer** for download and select **Continue**, as shown in Figure 2-4.

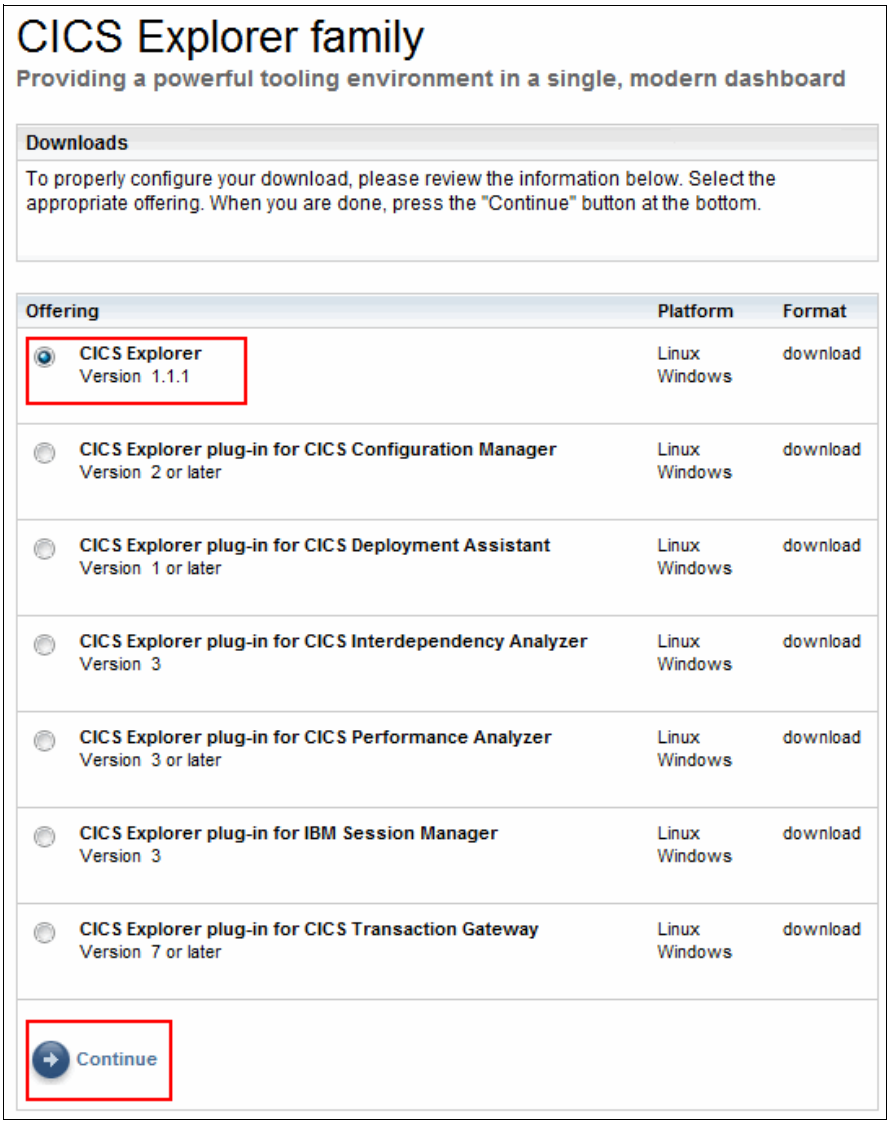


Figure 2-4 CICS Explorer download window: Select CICS Explorer for download

3. Select **CICS Explorer SDK** and start the download, as shown in Figure 2-5.

   After the download is complete, you see a compressed file called `cicsts_explorer_sdk_v1.1.x.x.zip`.

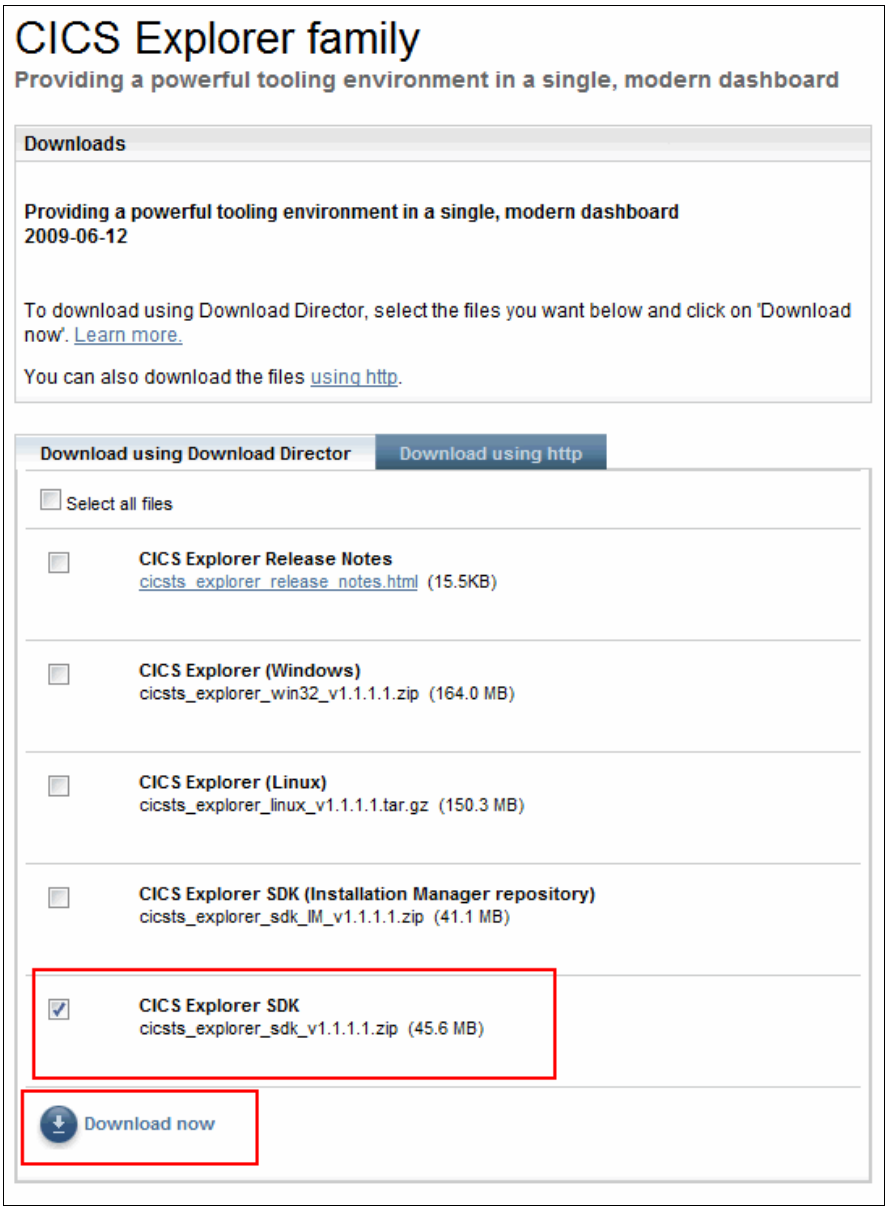


Figure 2-5 CICS Explorer download window: Select CICS Explorer SDK for download

4. Open the IBM Problem Determination Tools Studio.
5. Select **Help** → **Install New Software...** from the drop-down menu.
6. Click **Add** to specify the location of the compressed file that contains CICS Explorer SDK.

7. Select all of the CICS Explorer SDK features in the installation window, as shown in Figure 2-6.

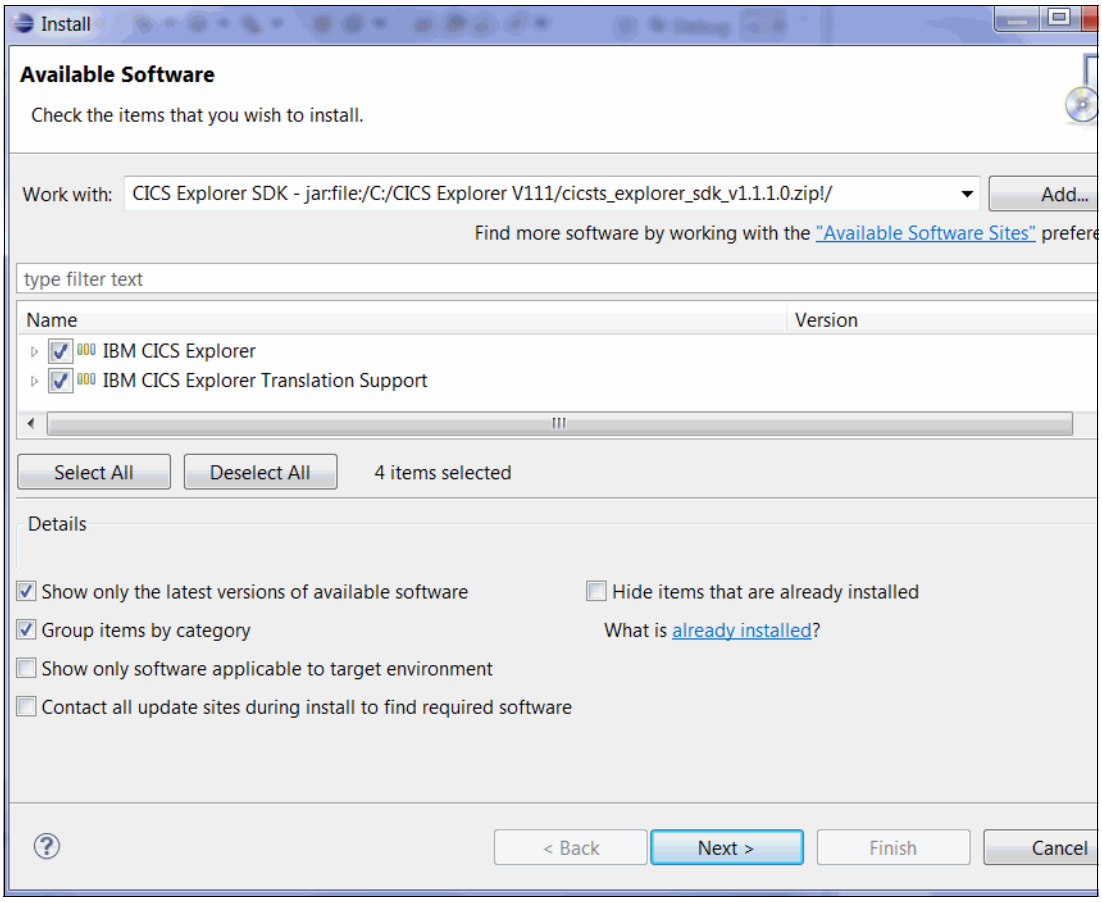


Figure 2-6 Specify a file that contains CICS Explorer SDK and install CICS Explorer features

8. Click **Next** to review the installation details.
9. Click **Next** to review and accept the license agreement.

After you accept the license agreement, click **Finish** to install CICS Explorer SDK capabilities.

2.2 IBM Problem Determination Tools SDK (P2)

You install the PD Tools plug-ins into a shell Eclipse environment of your choice by using IBM Problem Determination Tools SDK (P2). The IBM Problem Determination Tools SDK (P2) is not a stand-alone application. It is a set of the Problem Determination Tools plug-ins, which can be installed into your Eclipse environment. IBM recommends CICS Explorer V1.1.1 or later or IMS Explorer V2.1.1.2 or later tools for accessing views, data, and methods. In this section, we describe the steps that are used to install the PD Tools plug-ins into CICS Explorer and a stand-alone Eclipse environment.

The IBM Problem Determination Tools SDK (P2) contains the following plug-ins:

- Application Performance Analyzer for z/OS plug-in
- Debug Tool for z/OS plug-in (including DTSP and DTCN plug-ins)
- Fault Analyzer for z/OS plug-in
- File Manager for z/OS plug-in
- Workload Simulator for z/OS plug-in

2.2.1 Installation instructions for CICS Explorer

Complete the following steps to install the PD Tools plug-ins into a CICS Explorer environment:

1. Download a copy of CICS Explorer V1.1.1 or later from this website:

   `http://www.ibm.com/software/htp/cics/explorer/`

2. Select **CICS Explorer** for download, as shown in Figure 2-7.

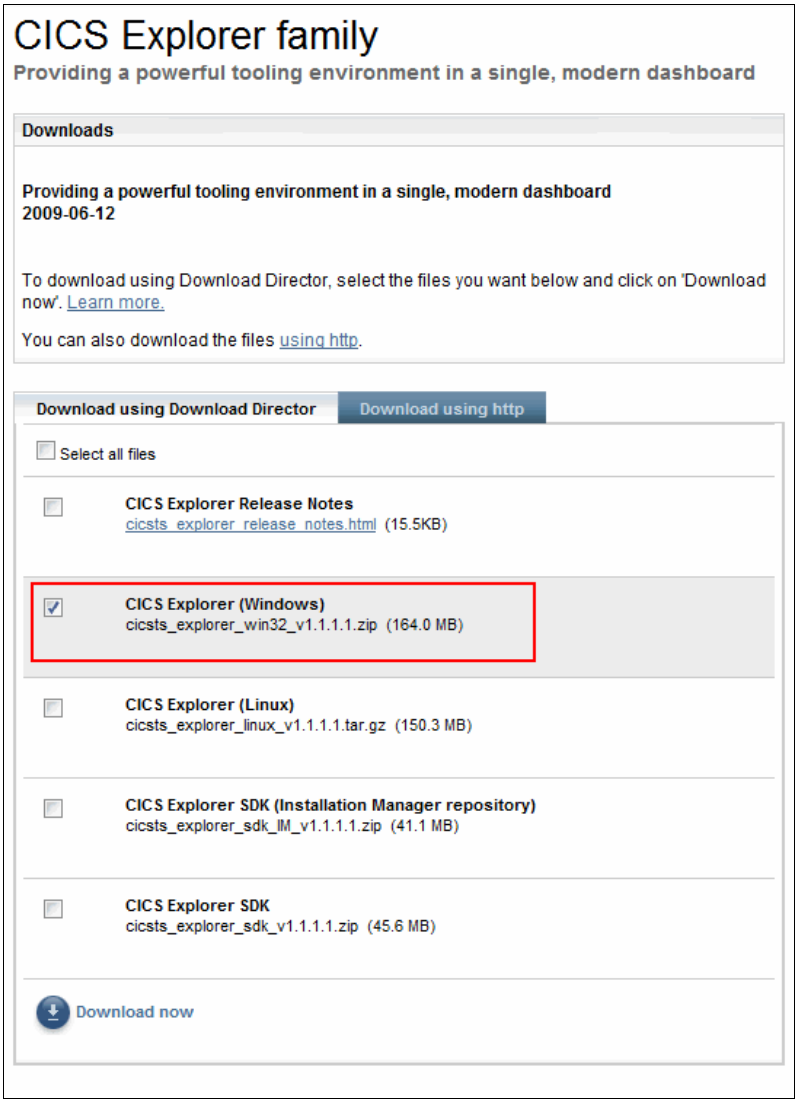


Figure 2-7 CICS Explorer download window

3. After CICS Explorer is downloaded, you see a compressed file called `cicsts_explorer_win32_v1.1.x.x.zip`. Extract this file into your folder of choice. After the file is extracted, you can delete the compressed file.
4. Start your CICS Explorer by double-clicking the **`cicsexplorer.exe`** file.
5. Download the IBM Problem Determination Tools SDK (P2) from this website:

   `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`.

   After the file is downloaded, you see the `PD_Tools_SDK_P2_V12xxxx.zip` file.

6. In your CICS Explorer, select **Help** → **Install New Software**... from the drop-down menu.
7. Click **Add** and specify the compressed file.
8. You see a list of the PD Tools plug-ins to install. Select the plug-ins to install and click **Next**, as shown in Figure 2-8.

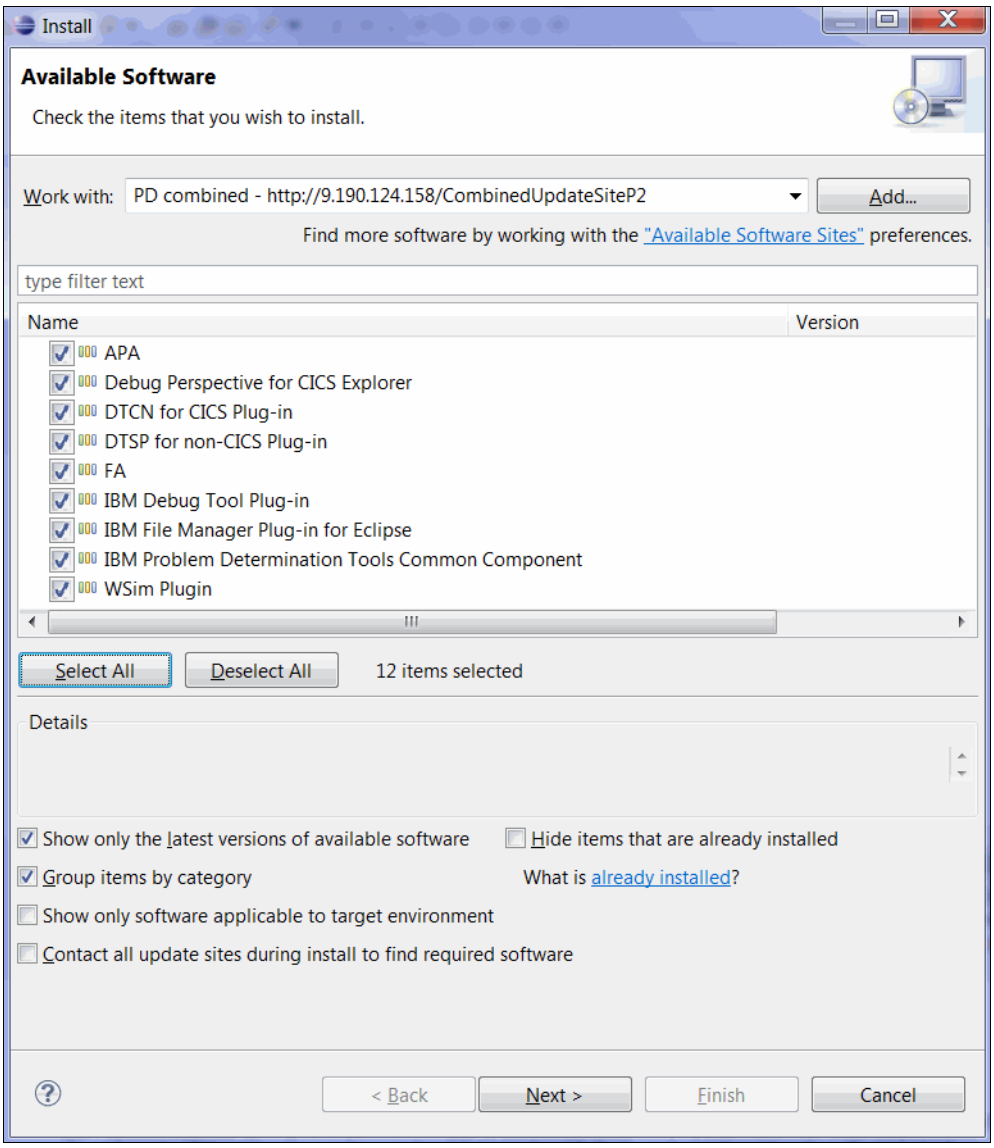


Figure 2-8 IBM Problem Determination Tools SDK (P2) installation window

9. Review the installation details and click **Next**.
10. After you review and agree to the license agreement, click **Finish** to complete the installation.

2.2.2 Installing IBM Problem Determination Tools SDK (P2) into your Eclipse environment

IBM recommends the use of CICS Explorer, IMS Explorer, or Rational Developer for System z as a shell to deploy the Problem Determination Tools plug-ins. Complete the following steps to add the PD Tools plug-ins into your own Eclipse environment:

Important: If you encounter a problem with the PD Tools plug-ins in your custom Eclipse environment, the problem must be able to be reproduced in one of the officially supported Eclipse environments to receive support.

1. Download a copy of the CICS Explorer SDK V111 or later from this website:

   `http://www.ibm.com/software/htp/cics/explorer/`
2. Select **CICS Explorer** for download and select **Continue**, as shown in Figure 2-4 on page 29.
3. Select **CICS Explorer SDK** to start the download, as shown in Figure 2-5 on page 30.

   After the download is complete, you see a compressed file called `cicsts_explorer_sdk_v1.1.x.x.zip`.
4. Start your Eclipse environment.
5. Select **Help** → **Install New Software...** from the drop-down menu.
6. Click **Add** to specify the location of the compressed file that contains CICS Explorer SDK.
7. Select all CICS Explorer SDK features in the installation window, as shown in Figure 2-6 on page 31.
8. Click **Next** to review the installation details.
9. Click **Next** to review and accept the license agreement.
10. After you accept the license agreement, click **Finish** to install CICS Explorer SDK capabilities.

    After the installation finishes, Eclipse restarts to complete the installation process.
11. Download the IBM Problem Determination Tools SDK (P2) from this website:

    `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`

    After the download is complete, you see the `PD_Tools_SDK_P2_V12xxxx.zip` file.
12. In your Eclipse environment, select **Help** → **Install New Software...** from the drop-down menu.
13. Click **Add** and specify the compressed file.
14. You see a list of the PD Tools plug-ins to install. Select the plug-ins to install and click **Next**, as shown in Figure 2-8 on page 34.
15. Review the installation details and click **Next**.
16. After you review and agree to the license agreement, click **Finish** to complete the installation.

2.2.3 Updating the IBM Problem Determination Tools SDK

The steps that are used to update the PD Tools SDK (P2) are the same steps that are described in 2.1.2, “Updating the IBM Problem Determination Tools Studio” on page 28.

2.3 The IBM Problem Determination Tools SDK (Installation Manager)

Rational Developer for System z is another supported environment for the Problem Determination Tools plug-ins. To install the PD Tools plug-ins, you must download a copy of the IBM Problem Determination Tools SDK (Installation Manager) and complete the steps that are shown in “Installation Instructions”. In the IBM Problem Determination Tools SDK (Installation Manager), the following plug-ins are included:

- Application Performance Analyzer for z/OS plug-in
- Fault Analyzer for z/OS plug-in
- File Manager for z/OS plug-in
- Workload Simulator for z/OS plug-in

2.3.1 Installation Instructions

You install the PD Tools plug-ins into a Rational Developer for System z V8.5 or later environment by using the IBM Problem Determination Tools SDK (Installation Manager). Complete the following steps to install the PD Tools plug-ins:

1. Download a copy of the IBM Problem Determination Tools SDK (Installation Manager) from this website:

   `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`

2. After the file is downloaded, you see a compressed file called `PD_Tools_SDK_IM_V12xxxx.zip`.
3. Start IBM Installation Manager.
4. Select **File** → **Preferences** from the drop-down menu.
5. In the Repositories tab, select **Add Repository** and specify the location of the compressed file on your computer.

6. Ensure that the repository location is selected, as shown in Figure 2-9.

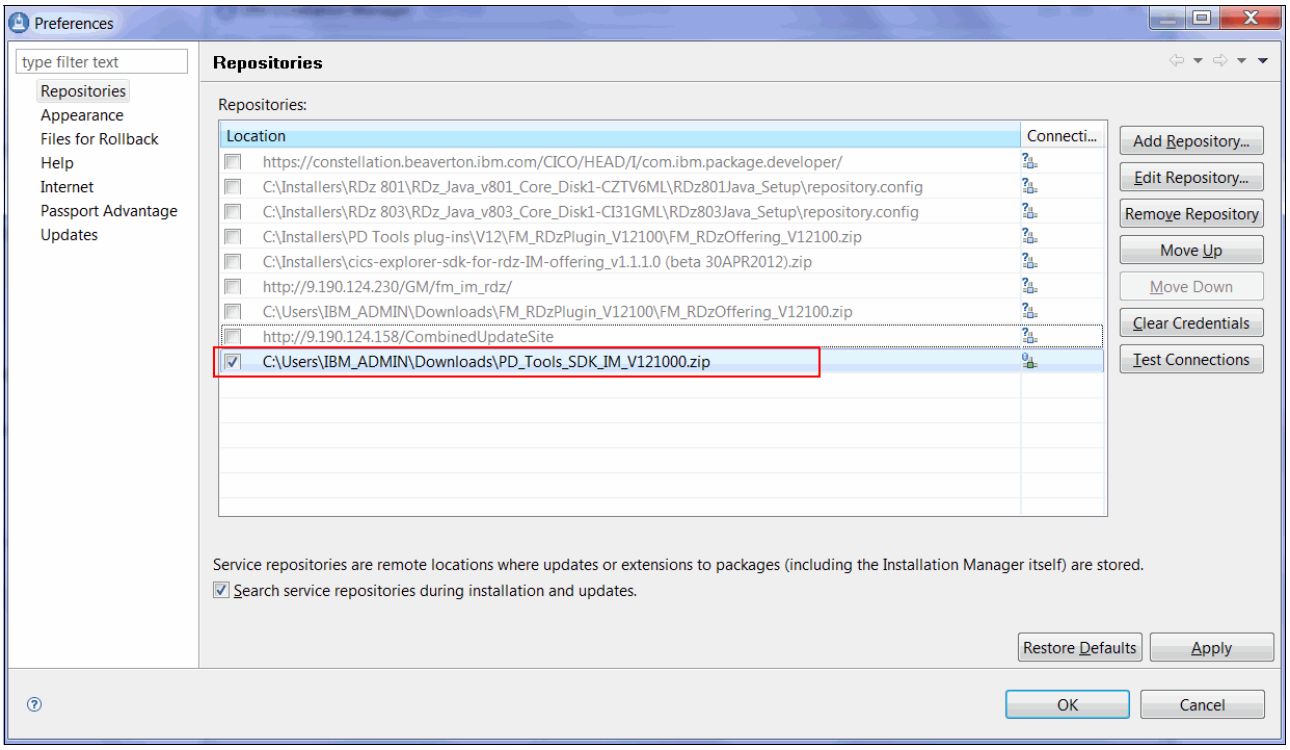


Figure 2-9 Installation Manager preference window for specifying repository locations

7. Click **OK** to close the Preference window.

8. Click **Install** in the IBM Installation Manager window, as shown in Figure 2-10.

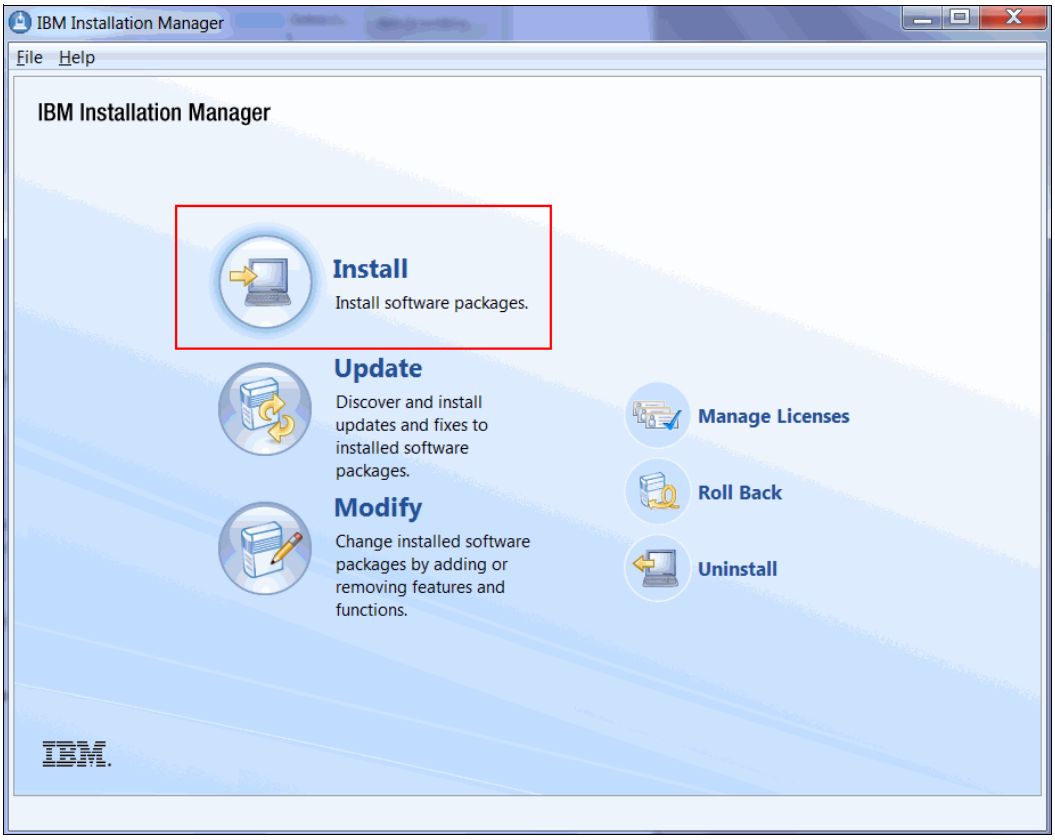


Figure 2-10 Selecting the Install option in the IBM Installation Manager window

9. Select the packages (PD Tools plug-ins) to install into your Rational Developer for System z environment, as shown in Figure 2-11.

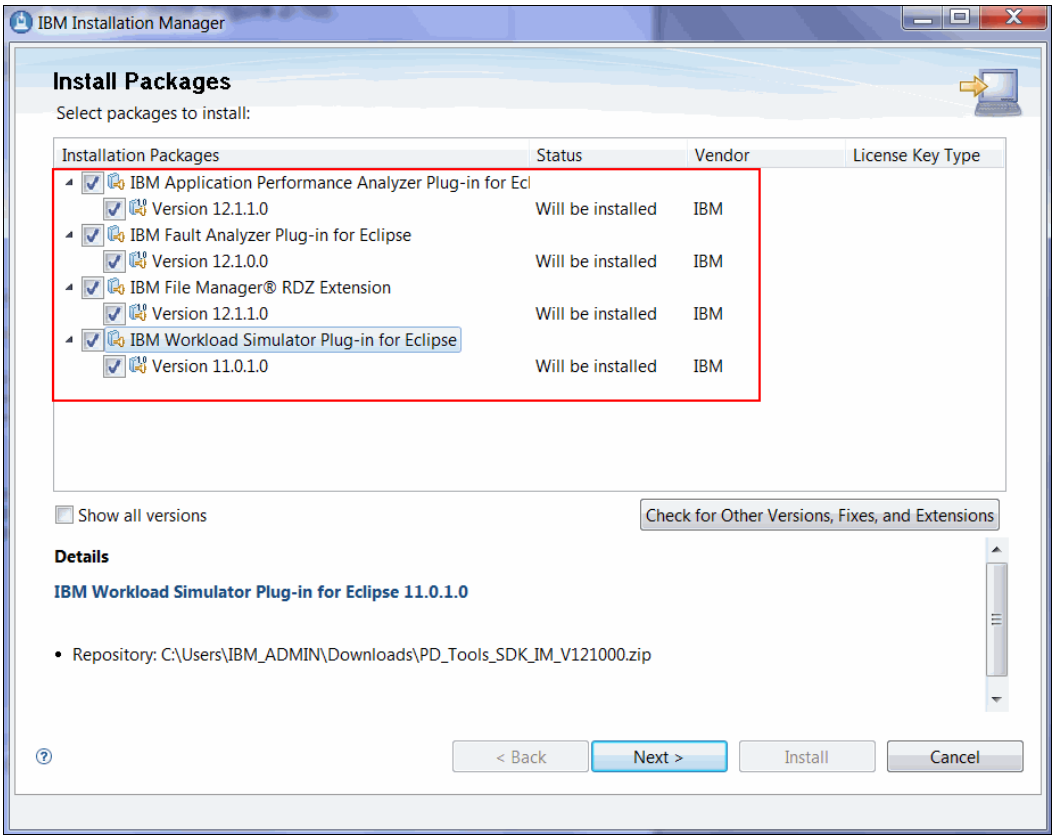


Figure 2-11 Selecting packages to install in the IBM Installation Manager window

10. Click **Next** to review and accept the licenses.
11. Click **Next** to identify the target environments in which the PD Tools plug-ins can be installed.
12. Select a target environment to install and click **Next**.
13. Review the list of packages to install and click **Next**.
14. Review the summary information and click **Install** to install the PD Tools plug-ins.

2.3.2 Updating the IBM Problem Determination Tools SDK (Installation Manager)

When a newer version of the IBM Problem Determination Tools SDK (Installation Manager) is available, you can update the installed plug-ins in your Rational Developer for System z environment by completing the following steps:

1. Download a copy of the IBM Problem Determination Tools SDK (Installation Manager) from this website:

   `http://www.ibm.com/software/awdtools/deployment/pdtplugins/`

2. After the file is downloaded, you see a compressed file called `PD_Tools_SDK_IM_V12xxxx.zip`.
3. Start IBM Installation Manager.

4. Select **File** → **Preferences** from the drop-down menu.
5. In the Repositories tab, select **Add Repository** and specify the location of the compressed file on your computer.
6. Ensure that the repository location is selected, as shown in Figure 2-9 on page 37.
7. Click **OK** to close the Preference window.
8. Select **Update** in the IBM Installation Manager window, as shown in Figure 2-12.

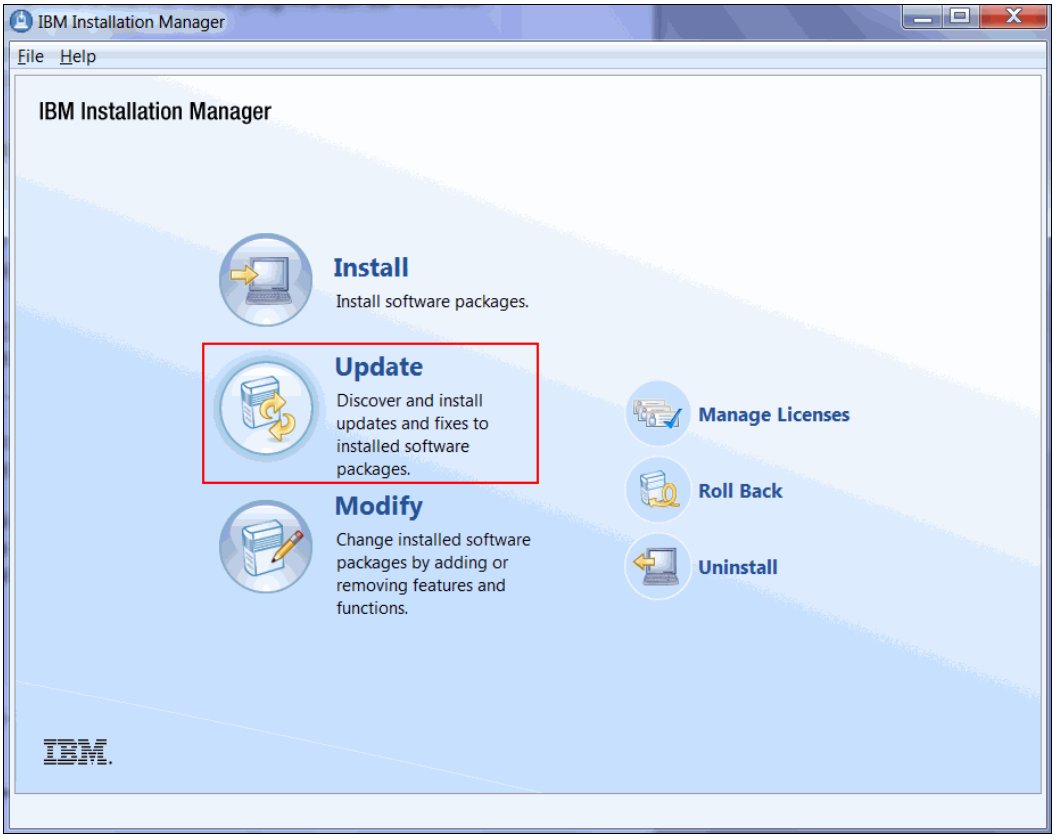


Figure 2-12 Selecting Update option in the IBM Installation Manager window

9. In the next window, select a target Rational Developer for System z environment on which you want to update the PD Tools plug-ins, as shown in Figure 2-13.

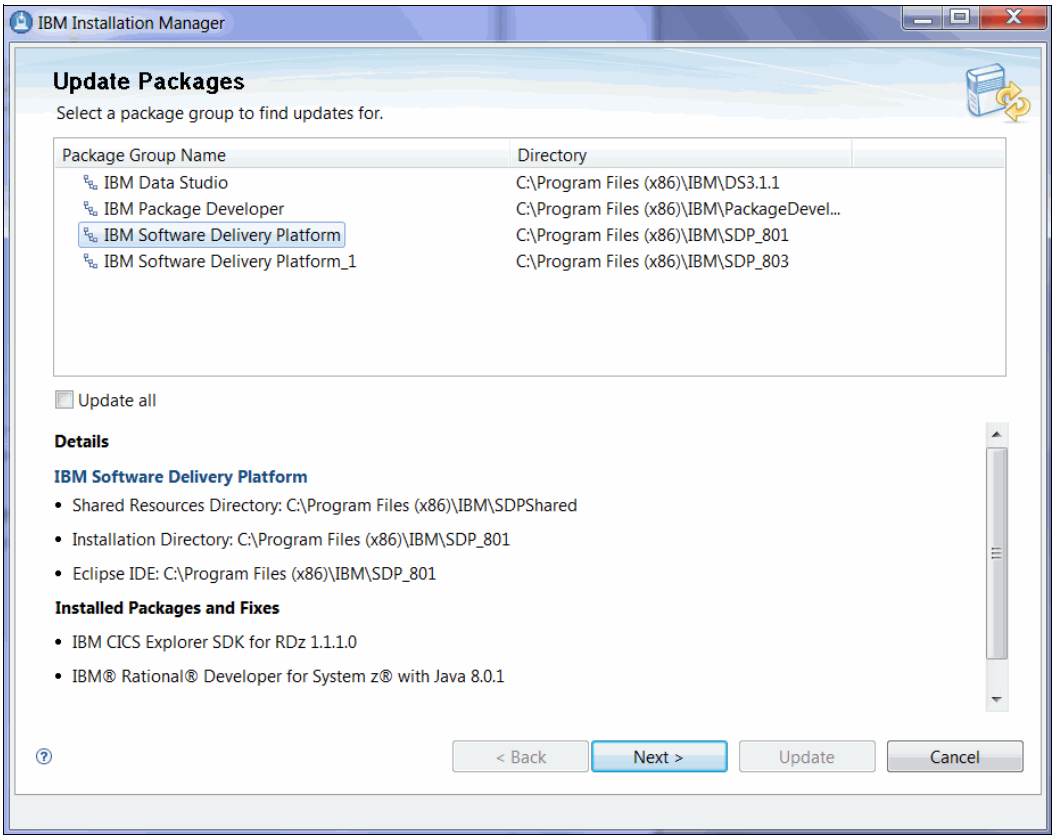


Figure 2-13 Selecting a target environment on the IBM Installation Manager window

10. Click **Next**. A process begins to determine whether any installed PD Tools packages require updates.

If any packages require an update, follow the remaining steps that are a part of this process to update the PD Tools plug-ins that are installed in your Rational Developer for System z environment.

2.4 Introduction to the Problem Determination Tools Common Component and z/OS Explorer Host Connection framework

An overview of the components that makes the PD Tools plug-ins work is shown in Figure 2-14.

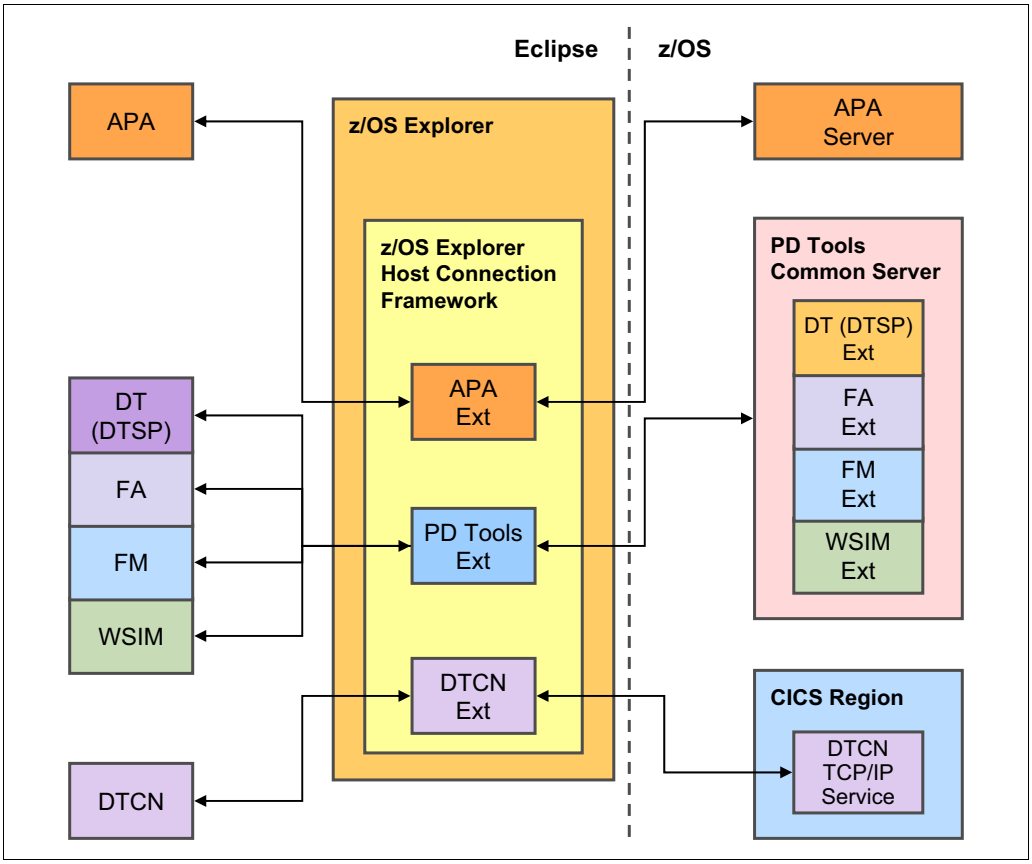


Figure 2-14 Overview of the IBM Problem Determination Tools plug-ins and the host configuration

The z/OS Explorer is a common feature that is available in various IBM products, such as CICS Explorer and IMS Explorer. This feature provides fundamental functionality to interact with z/OS systems from an Eclipse environment. Among many features of the z/OS Explorer, the PD Tools plug-ins relies on its Host Connections framework to manage connections, user IDs, and passwords.

As shown in Figure 2-14, the PD Tools plug-ins define the following connection types on the host connection framework:

- Problem Determination Tools for z/OS connection type for connecting the Fault Analyzer plug-in, File Manager plug-in, Workload Simulator plug-in, and Debug Tool DTSP plug-in.
- Application Performance Analyzer connection type for connecting the APA plug-in.
- Debug Tool DTCN connection type for connecting the DTCN plug-in.

Figure 2-15 shows the z/OS Explorer's Host Connections view in which you find three connection types that are required by the PD Tools plug-ins.

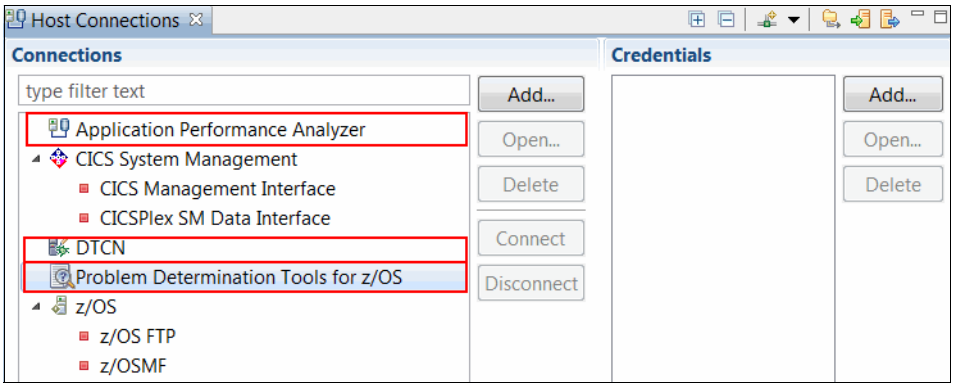


Figure 2-15 z/OS Explorer's Host Connection view

Problem Determination Tools Common Server is a stand-alone application that runs on a z/OS system. The server is extensible, which means that you can extend the functionality of the server by modifying the configuration file of the server, depending on which PD Tools plug-ins are required. As shown in Figure 2-14 on page 42, Debug Tool for z/OS, Fault Analyzer for z/OS, File Manager z/OS, and Workload Simulator for z/OS provide extensions to this server to serve their corresponding Eclipse plug-ins.

Application Performance Analyzer's (APA) plug-in and server are required to use the APA plug-in.

DTCN plug-in and DTCN TCP/IP Service are required to use the DTCN plug-in.

2.4.1 Establishing a connection to the PD Tools server

Complete the following steps to establish a connection to the PD Tools server:

1. Select a connection type and click **Add**... (for example, Problem Determination Tools for z/OS).
2. Specify the host name and the port number of the server, as shown in see Figure 2-16 on page 44.

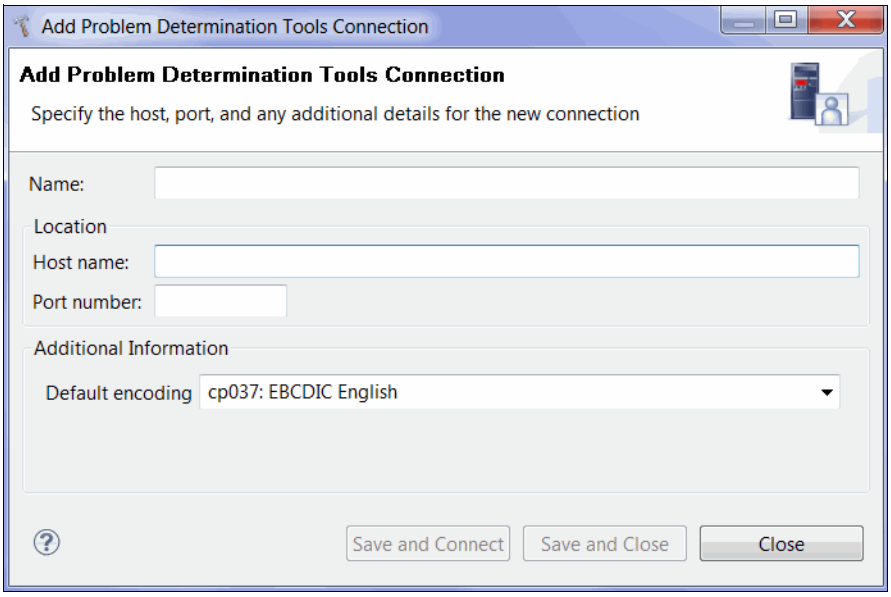


Figure 2-16 Host Connections dialog for specifying connection details

3. After the connection information to a server is created, you must specify your user name for the connection. As shown in Figure 2-17, you can create your credential (USER1 in the figure) in the Credentials section.

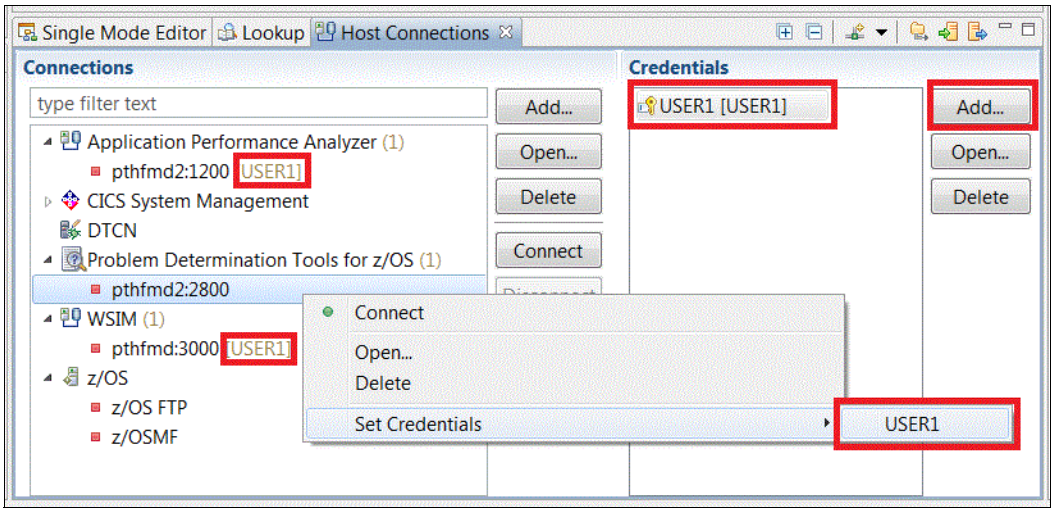


Figure 2-17 Setting a password for a connection in the Host Connections views

After your credential is created, you can associate the credential to any connections that are defined in the Host Connections view. For example, USER1 user ID is assigned to an instance of the APA connection type, the PD Tools connection type, and the DTCN connection type.

2.4.2 Sharing Host Connections information

The steps that are described in “Establishing a connection to the PD Tools server” on page 44 require individual users to establish connection-required servers manually, which can be time-consuming and error-prone. Instead, the Host Connections framework provides a mechanism for sharing connection information easily among users.

Complete the following steps to share connection information:

1. Click **Load Connections**, which is at the upper right corner of the Host Connections view, as shown in Figure 2-18.

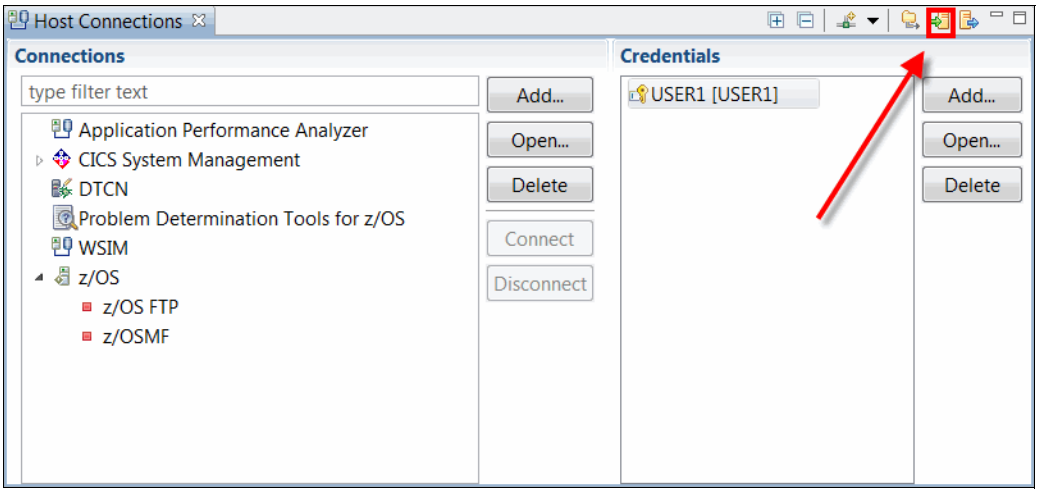


Figure 2-18 Invoke the Host Connections import connection information window

2. In the Load Connections dialog, specify the location of a file that contains the connections information. This file often is maintained by a system administrator. The file can be found in a local file or on a remote server, as shown in Figure 2-19.

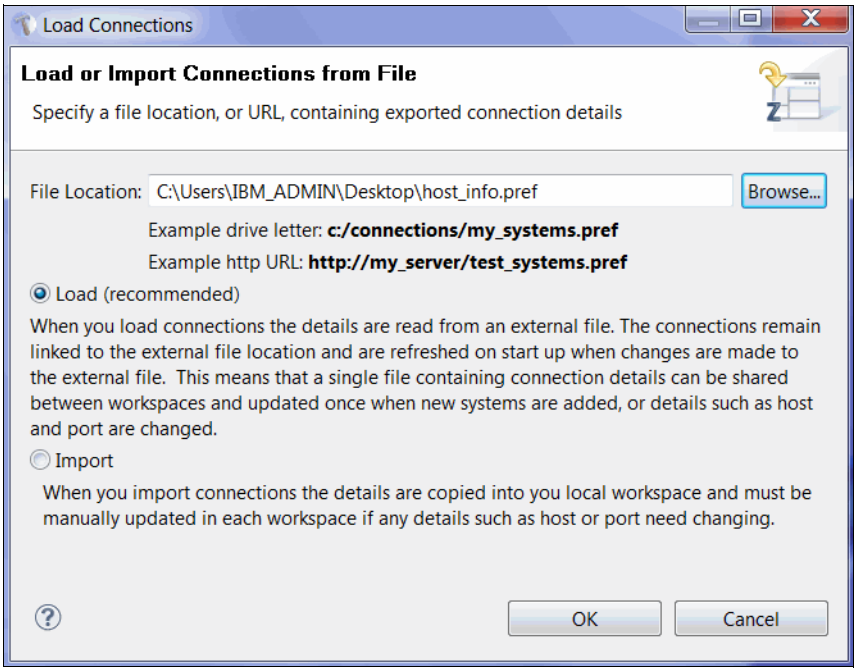


Figure 2-19 Load Connections window for importing the connections information

3. Click **OK** to load the connections information. As shown in Figure 2-20, several pieces of connection information (highlighted in red) are loaded into the view.

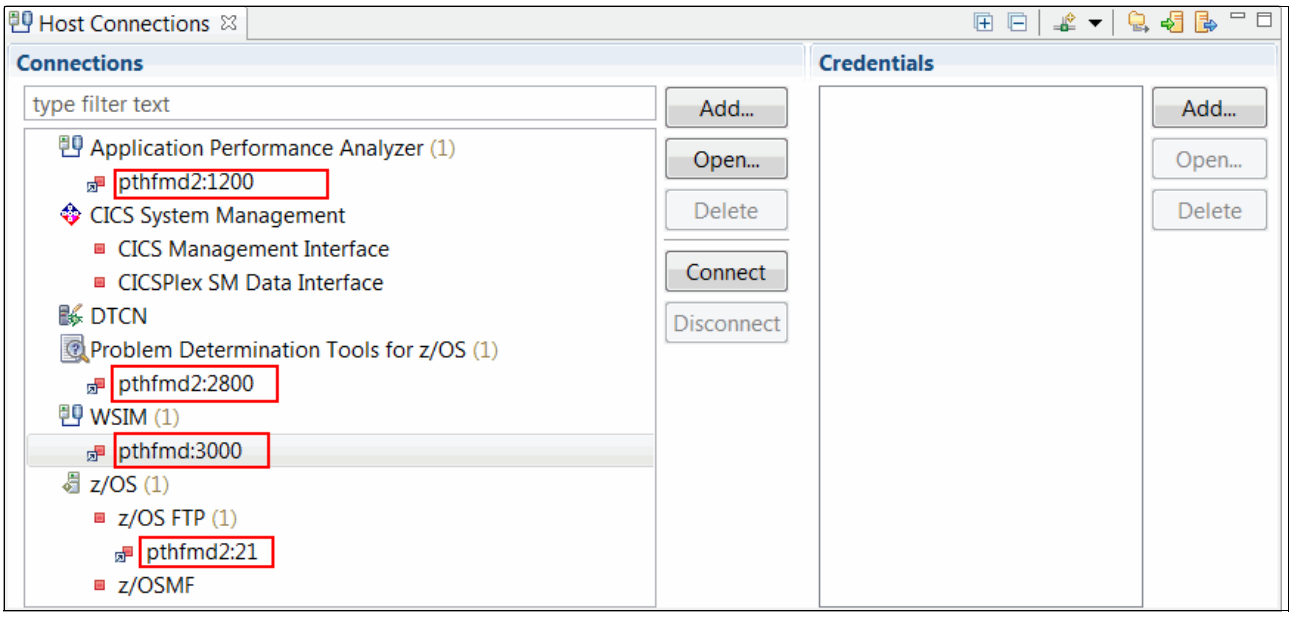


Figure 2-20 Imported connection information

3

Interfaces with IBM Problem Determination Tools: 3270

In this chapter, we provide an overview of the interfaces that are available for interacting with the IBM Problem Determination Tools through a 3270 display.

3.1 Problem Determination Tools interfaces: Overview

In this section, we describe the Problem Determination Tools interfaces and features that are available.

3.1.1 ISPF

To use the Problem Determination Tools with Interactive System Productivity Facility (ISPF), you must ensure that the appropriate data sets are allocated. You must also ensure that one or more ways to start each of the installed products are provided.

3.1.2 CICS

Fault Analyzer and File Manager can be started under the Customer Information Control System (CICS).

Fault Analyzer

Fault Analyzer uses a component to display ISPF panels that can allow it to operate as a CICS transaction to view history files and perform interactive reanalysis. This capability under CICS does not use TSO. It is primarily intended for users who might not have TSO logon capability on an MVS image, but must review and analyze history file information about that MVS image. The appearance of the display is similar to that of the equivalent ISPF display.

File Manager

File Manager for CICS (FM/CICS) is a powerful set of utility functions for editing, browsing, printing, and altering the status of CICS resources. The CICS resources that are supported are files, temporary storage queues, transient data queues, and data tables. If you have the authority, you can also modify the status of the CICS resources. FM/CICS incorporates much of the functionality of File Manager for z/OS (base) into the CICS environment.

3.2 Getting started

For most customers, starting a Problem Determination Tools session is done through menu selection from an ISPF panel. The option of using a TSO command also is available.

3.2.1 Application Performance Analyzer

In this section, we describe the features of the Application Performance Analyzer (APA).

ISPF

Figure 3-1 shows the Observation List panel, which is the initial window of the APA. Existing entries, which were previously stored in the Measurements data set, are displayed; by default, the most recent entry is at the top. The list can be reordered and filtered (often by owner ID). From this panel, you can perform the following tasks:

- Schedule a new measurement session, for a running or future job
- Review previously collected measurements

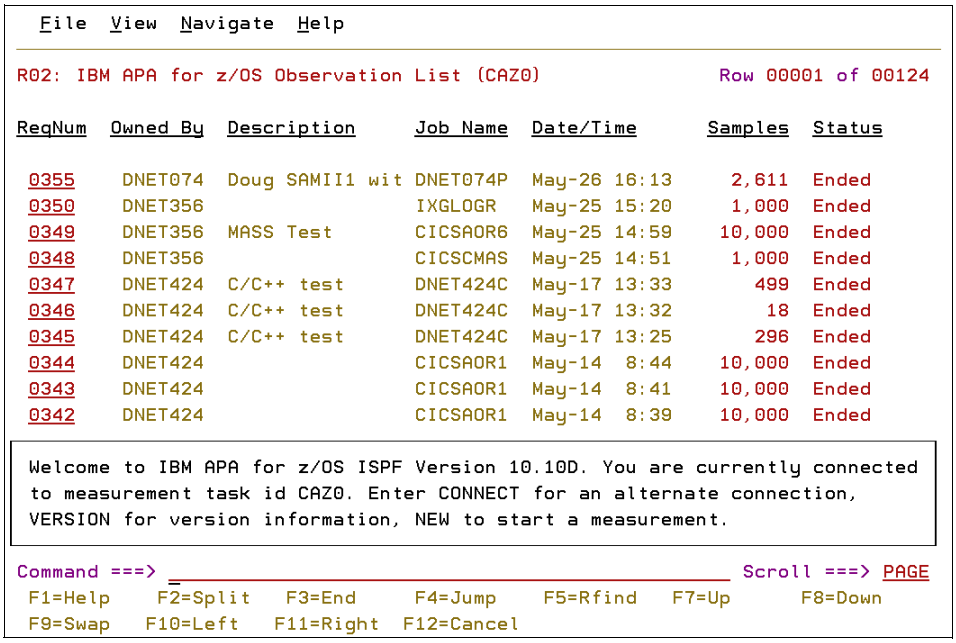


Figure 3-1 Initial APA display

Use the **SHOW** and **HIDE** primary commands to display and hide the full list of commands that are available.

Entering a "/" in the column on the left of one of the line entries causes a window to show that contains a list of all the line commands you can use for that entry. You also can select the commands from the list.

CICS

There are no CICS considerations for the APA.

3.2.2 Debug Tool

In this section, we describe the features of the Debug Tool.

ISPF

Debug Tool features a number of utilities, which are listed in the initial window upon selecting Debug Tool from your ISPF display, as shown in Figure 3-2.

```
--------------------------- Debug Tool Utilities ---------------------------
                                                                More:     +
0  Job Card
   Create Job Card image.

1  Program Preparation
   Convert, compile, assemble or link edit program.

2  Debug Tool Setup File
   Manage setup files and start debug session in TSO foreground or batch.

3  Code Coverage
   Measure code coverage in programs.

4  IMS TM Setup
   Update Language Environment run-time options in IMS. Create message region.

5  Load Module Analyzer
   Analyze load modules and each CSECT in the load module.

6  Debug Tool User Exit Data Set
Option ===> _____________________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-2 Utilities selection

An actual debug session is not started from an ISPF menu selection. The options that are selected when a program is compiled and when it is run control the start of a debug session.

Figure 3-3 on page 51 shows the initial display that is shown when a debug session is entered. It provides an interactive interface that includes the following windows that enable single-step debugging, dynamic patching, and breakpoint setting:

- A monitor window shows the status of items you select, variables, and registers. You can view, monitor, and alter application variables or storage in real time.
- A source window shows the program code and highlights the statement that is run. In the prefix area of this window, you can enter commands to set, display, and remove breakpoints.
- A log window records and shows your interactions with Debug Tool and can show program output. The information that you see in this window is included in the log file.
- By using the memory window (which is swappable with the log window), you can show and scroll through sections of memory. You also can update memory by replacing existing data with new data. The memory window tracks addresses for easier navigation.

By using the Debug Tool source window (as shown in Figure 3-3), you can monitor application code while an application runs. You also can debug applications that are written in a mix of COBOL, C, C++, PL/I, or Java languages without leaving the tool. You can include Assembler programs in this mix and, by using the disassembly view, you can debug programs that are compiled with the NOTEST compiler option or applications that include other languages. You also can use commands to dynamically patch or alter the value of variables and structures and control the flow of an application.

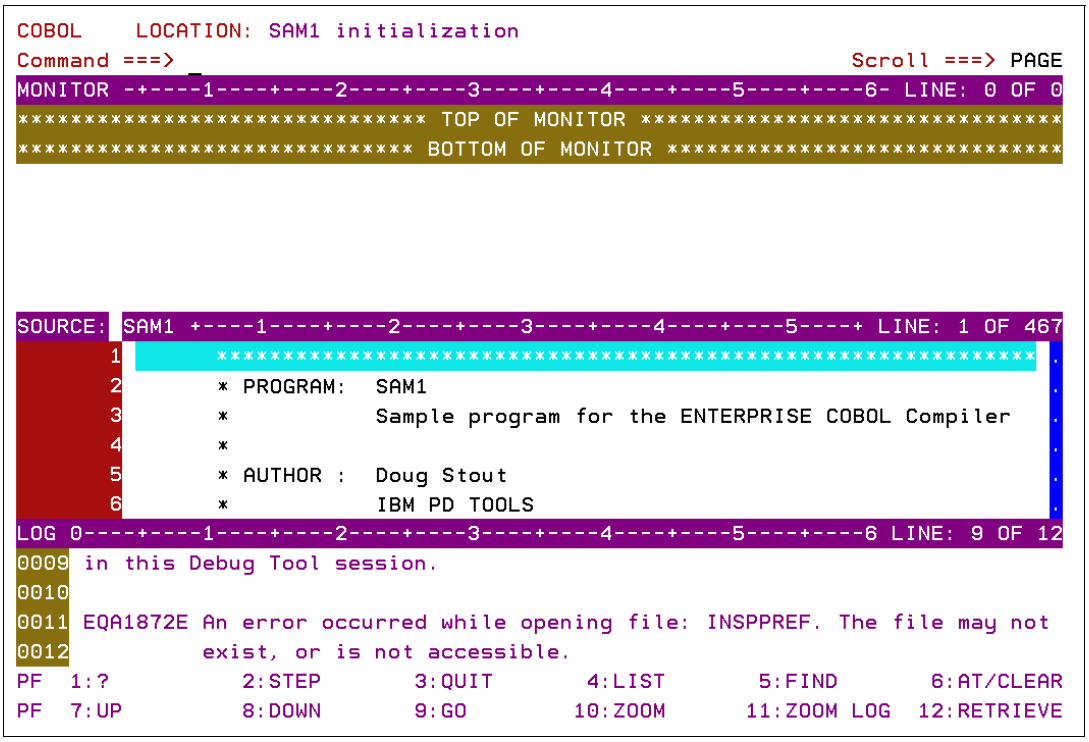


Figure 3-3 Debug session starts

CICS

The Debug Tool Control utility is a CICS transaction (transaction ID DTCN), with which the user identifies the CICS programs to debug. When the required debug session starts, the initial display is show (see Figure 3-3).

3.2.3 Fault Analyzer

In this section, we describe the features of the Fault Analyzer.

ISPF

Figure 3-4 shows the Fault Entry List panel, which is the initial display of the Fault Analyzer when it is selected from an ISPF menu. Existing entries, previously stored in a Fault History data set, are shown; by default, the most recent entry is shown at the top. Views can be used if the containing data set is defined in the Fault Analyzer options data sets. The list can be reordered and filtered by column and wildcard specification. To work with entries from a different history file, type over the displayed data set name and press Enter. History files can be managed through the Fault Entry List panel. Fault entries can be copied and moved to other history files.

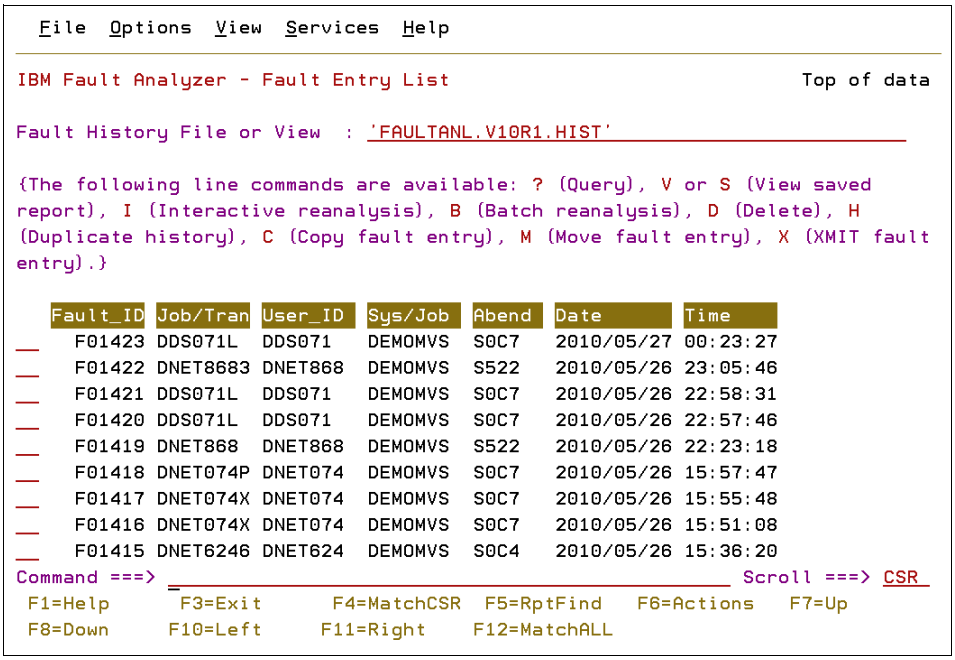


Figure 3-4 Initial display

Interactive reanalysis, initiated from this panel, runs under ISPF. By using this reanalysis, and you can browse through a formatted, structured view of a fully detailed reanalysis. By using this Fault Analyzer mode, you can view working storage and control blocks at the time the dump was written. The ISPF interface has many point-and-click fields for easy navigation through the interactive reports. Interactive reanalysis can also be performed against system dumps. CICS abends can be viewed only by using interactive reanalysis.

CICS

Required CICS resource definitions, including the definition of the transaction that is required to start Fault Analyzer, are provided in a sample job that is supplied with the code. These definitions must be installed in any CICS system in which support for Fault Analyzer is required. The default transaction identifier is initial domain identifier (IDI), though this default can be changed.

The CICS interface is mostly identical to the ISPF interface. For more information, see the chapter “Using non-ISPF interfaces to access Fault Analyzer history files” of the *IBM Fault Analyzer for z/OS User’s Guide and Reference*, SC19-3671-01.

Real-time

Fault Analyzer provides abend invocation exits for CICS and batch, in Language Environment (LE) and non-LE environments.

The software includes exit programs (for CICS, IBM Language Environment, and z/OS systems) that it adds to the normal failure processing for these environments. When an application failure occurs in any of these environments, the Fault Analyzer exit starts real-time analysis. After failure processing, you can view the analysis report in the job output or through the Fault Analyzer ISPF interface.

Batch

Batch reanalysis generates a new analysis report. This report is based on the dump and information that is gathered in real time, but with potentially different options that are specified, or with program source information that is made available.

3.2.4 File Manager

In this section, we describe the features of the File Manager.

ISPF

Figure 3-5 shows the Primary Option panel, which is the initial display of the base component of File Manager. Similar initial panels are shown as the entry point to each of the three other File Manager components.

```
 Process    Options    Help
------------------------------------------------------------------------------
File Manager                  Primary Option Menu
Command ===> _
------------------------------------------------------------------------------
0  Settings       Set processing options                 User ID . : KENICHI
1  View           View data                              System ID : FMD2
2  Edit           Edit data                              Appl ID . : FMN
3  Utilities      Perform utility functions              Version . : 12.1.0
4  Tapes          Tape specific functions                Terminal. : 3277
5  Disk/VSAM      Disk track and VSAM CI functions       Screen. . : 1
6  OAM            Work with OAM objects                  Date. . . : 2012/07/19
7  Templates      Template and copybook utilities        Time. . . : 20:19
8  HFS            Access Hierarchical File System
9  WebSphere MQ   List, view and edit MQ data
10 CICS           FM/CICS
X  Exit           Terminate File Manager




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 3-5 Base Primary Option menu

The following functions are available through ISPF panels:

- View and Edit data sets with DBCS support
- Create data sets of different types and initialization settings
- Display user storage
- Display load modules
- Compare data sets
- Display WebSphere MQ managers and queues
- Display raw disk layout information
- OAM object view, edit, and copy and conversion
- UNIX System Services (USS) and hierarchical file system (HFS) access
- File Manager (FM) and Customer Information Control System (CICS) access

A full list of functions is available in the “Panels and Fields” chapter of the *IBM File Manager for z/OS User’s Guide and Reference,* SC19-3675-00.

CICS

The required CICS resource definitions, including the definition of the transaction that is required to start File Manager, are supplied with the code. These definitions must be installed in any CICS system where support for File Manager is required. The default transaction identifier is FM, but this default can be changed.

3.3 Interface customization

In this section, we describe the features of Interface customization.

3.3.1 Application Performance Analyzer

You can use the PREF (PREFerences) command to set the preferences for general display settings. Include a slash “/” beside an option to select it, as shown in Figure 3-6.

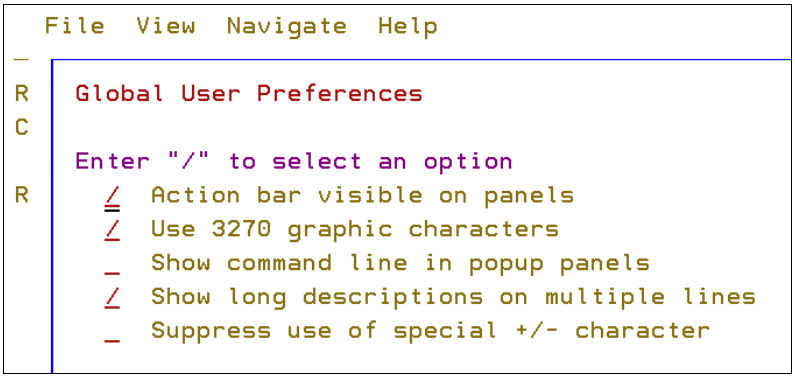


Figure 3-6 Setting APA preferences

3.3.2 Debug Tool

After a debug session is started, commands and special panels can be used to perform the following tasks:

- Hide or display a window
- Change the sizes of windows
- Change PF key settings
- Change the window arrangement
- Change the colors used
- Change Debug Tool session settings
- Change profile settings
- Save customization settings to file and restore between sessions

For more information about performing these tasks, see the “Customizing your full-screen session” chapter of *IBM File Manager for z/OS User’s Guide and Reference,* SC19-3675-00.

3.3.3 Fault Analyzer

The Fault Analyzer 3270 interface can be customized by selecting the View menu, as shown in Figure 3-7.

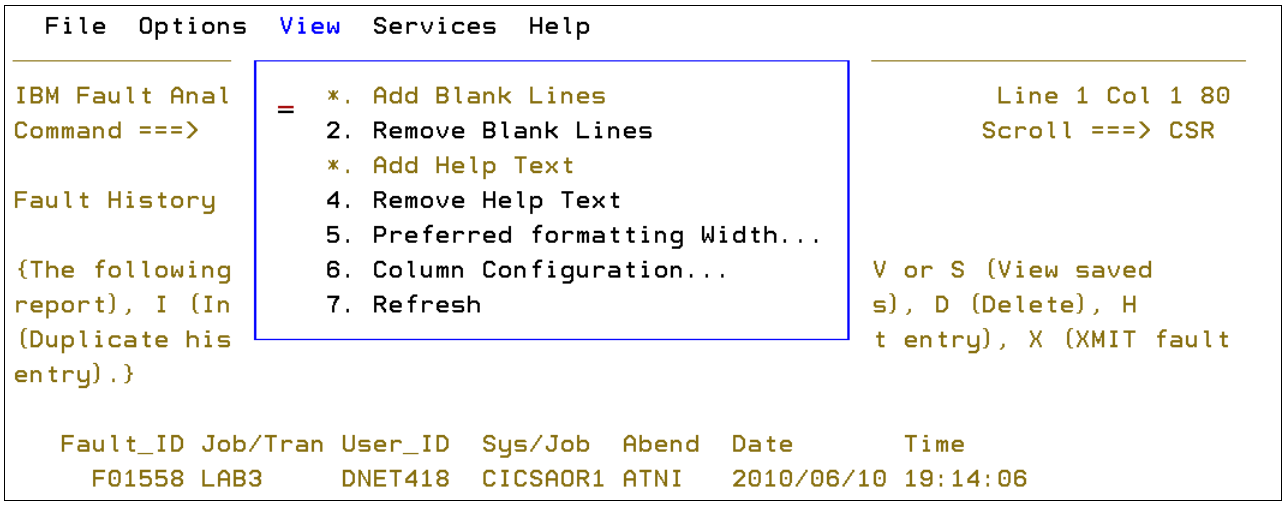


Figure 3-7 Setting FA preferences

The generated reports can be modified by using a formatting user exit. A formatting user exit uses an HTML-like language to change the layout of presented information.

3.3.4 File Manager

Many of the processing operations that are performed by File Manager use default values that can be set from within the File Manager application, as shown in Figure 3-8. By adjusting these values, you can customize File Manager so that their behavior is best-suited to your needs. Your settings for these options are stored in your ISPF profile and are started when you log in, regardless of which workstation you use. You can update these default values by accessing the relevant processing option panel, in one of the following ways:

- From the File Manager Primary Options menu for any of the components, select **Options → 0 (Settings)**.
- From any File Manager panel, use the Options menu to select the required processing options type.

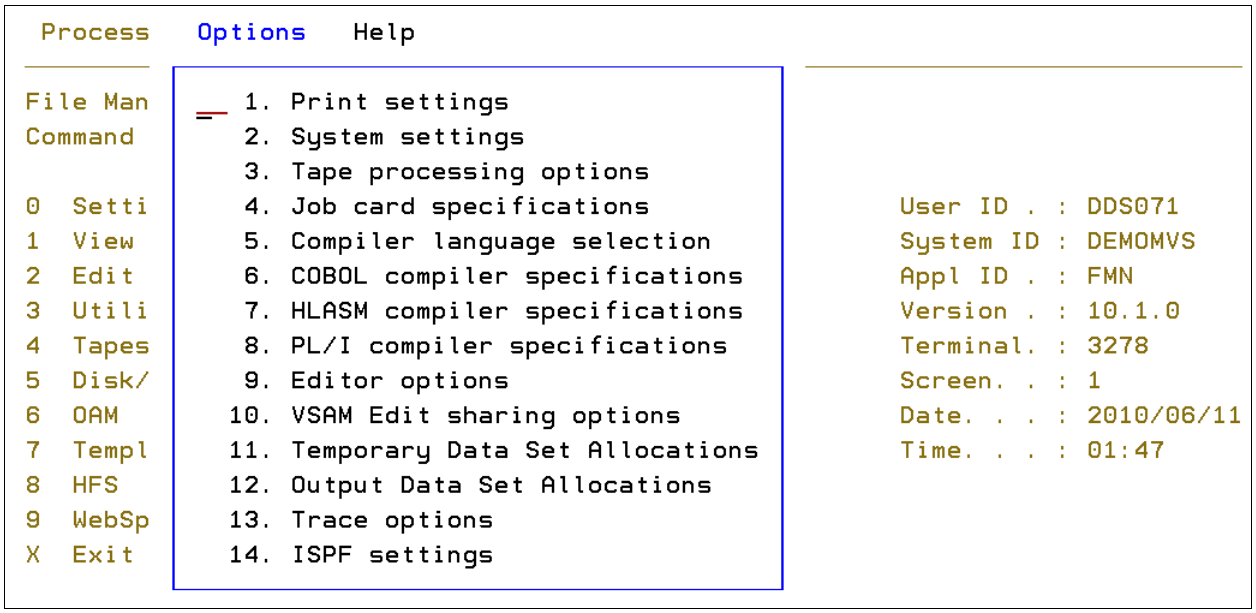


Figure 3-8 Setting FM preferences

3.4 Workload Simulator

In this section, we describe Workload Simulator (WSim), a terminal and network simulation tool that can be used to evaluate network design, perform and automate testing, and determine system performance and response time.

By using WSim, you can evaluate and test systems without the need for real terminals and terminal operators. WSim can be used to simulate actions of several types of applications and terminals. The simulated resources communicate among themselves and with the real teleprocessing system (called *system under test*) as though they physically exist. The system under test does not have to be modified.

3.4.1 Overview

Workload Simulator is an automated testing tool that can simulate terminals and other network devices and associated network traffic, and report the status and results of tests. It can be used to perform several types of tests. Workload Simulator features the following components:

- Batch utilities:
  - Capture data from live sessions
  - Prepare scripts
  - Run simulation tests
  - Generate reports
- WSim ISPF panels:
  - Run utilities online
  - Generate JCL
  - Run simulation tests interactively
  - Review test results
- WSim Test Manager ISPF application:
  - Simplifies and automates test process
  - Organizes tests by maintaining projects, test cases, network definitions, documentation, reports, and logs

A general overview of the WSim use context is shown in Figure 3-9.

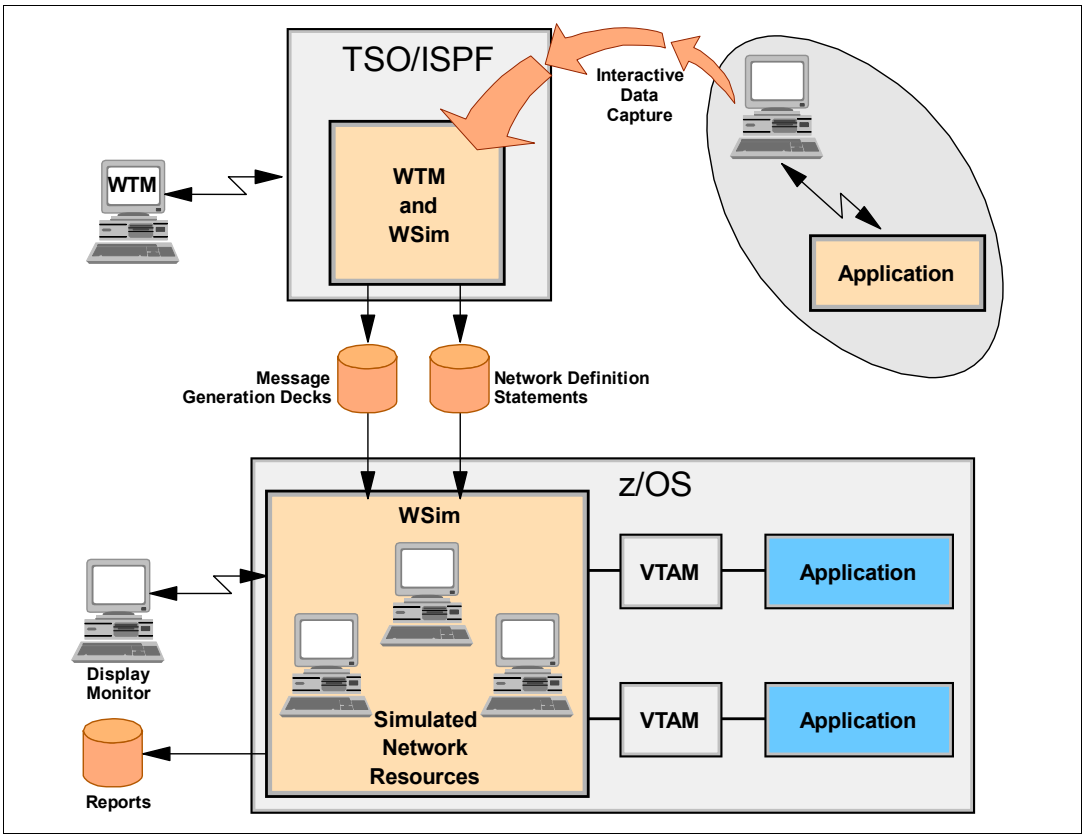


Figure 3-9 WSim use context

Use of WSim

To use WSim, you must prepare the following types of information:

- Network definition statements that describe the configuration of the simulated network
- Message generation decks that send and receive messages

Network definition statements and message generation decks form a script, which WSim uses to send messages to the system under test. WSim collects and records the information that is received from the system under test. It also uses this information to determine which messages to send back to the system under test.

WSim enables the system under test to operate (to a certain degree) as it does under actual conditions.

The process to conduct a test includes the following steps:

1. Plan the test:
   a. Define the objectives.
   b. Prepare the test plan.
2. Configure the system:
   a. The actual system that is used to run WSim (physical configuration).
   b. The simulated network (logical configuration).
3. Prepare testing scripts:
   a. Prepare network definitions for the network to be simulated.
   b. Prepare message generation decks.
4. Run the test.
5. Analyze the results from WSim.

Planning is an important task in this process. You should view the planning process as an ongoing task and be prepared to refine the plan until wanted results are achieved. Every test, especially when you begin by using WSim, should start with a small sample network definition and a small simple message generation deck. After successful runs, more refined system configurations and more complicated scripts should be prepared until the simulation is done for the complete network to be tested.

Resources WSim can simulate

WSim can simulate the following types of resources:

- System Network Architecture (SNA) logical units (LU) that are running as Virtual Telecommunications Access Method (VTAM®) applications
- Common Programming Interface for Communications (CPI-C) transaction programs (TP)
- Transmission Control Protocol/Internet Protocol (TCP/IP) clients who are using Telnet 3270, 3270E, 5250, Network Virtual Terminal (NVT), File Transfer Protocol (FTP), or Simple TCP and UDP protocols that are attached to the TCP/IP network via the IBM TCP/IP for Multiple Virtual Storage (MVS) product

Testing with WSim

WSim can be used to conduct the following types of tests:

- Function
- Regression
- Performance
- Stress
- Capacity planning

Function testing often is used to test a particular function of the system and answer the question, “Is it working correctly?” WSim can be used to test functions, such as new application transactions, logon and logoff procedures, error transactions, new hardware additions, and new software products. The scripts that are used in functional tests can be saved and reused for regression or stress tests.

Regression testing verifies that old functions operate correctly after the addition of new functions, or after any other changes to the existing system. This testing also answers the question, “Is it still working correctly?” The use of WSim for regression tests includes the following advantages:

- Scripts are repeatable. After the scripts are created, they can be reused many times until the tested functions are changed.
- WSim can be run automatically. Execution parameters and operator commands that control WSim operation, including ending the simulation after all of the test cases are completed, can be included in scripts.

Performance testing includes taking measurements, changing parameters, and taking measurements again. It answers the question, “How well does it perform?” WSim can be used to report terminal response times and provides the possibility to create a controlled, repeatable transaction load for the system under test.

Stress testing is performed when you must find problems in interactions and resource contentions. By loading the system under test with high transaction rates, you can answer the question, “What will break first?” This type of test is almost impossible to conduct without a special tool. WSim can generate controlled message traffic at controlled rates.

Capacity planning helps to predict how the system under test behaves when new resources are brought online or when one or more of the existing resources are overused. This type of test helps to determine whether the system under test still performs adequately under predicted increased load. This test also answers the question, “What happens if this many resources are added?” WSim can drive the system under test with a higher than normal transaction rate and simulate more terminals or different types of terminals.

When performance, stress, and capacity planning tests are conducted, WSim should be run on a separate host from the system under test to avoid affecting the results.

3.4.2 System configuration

The following terms are used in this chapter:

- Logical unit (LU)

  The LU is a port through which a user accesses an SNA network to communicate with another user or the system services control point (SSCP).

- Transaction program (TP)

  In WSim, the TP is any program that uses LU6.2 communications protocols to communicate with another program. WSim implements TPs by using CPI-C.

- Session

  A session is a logical connection that enables two network-addressable units to communicate with each other, such as an LU-LU, or an SSCP-LU session. Each half of a session is a half-session.

The network to be simulated and the system to be used to run WSim must be configured before testing. Configuration of the network that contains resources to be simulated by WSim and the real system to be tested (the system under test) is known as a *logical configuration*. For each logical configuration, a specific *physical configuration* must be used, which is the configuration of the real system that is used to run WSim. Resources of a physical configuration include a host processor, system software, application software, and WSim.

Physical configurations

WSim can operate in one of the following basic physical configurations:

- VTAM and CPI-C application configuration

  This configuration is used to simulate LUs in the same subarea as VTAM. An LU can have a session with any other LU with which that VTAM allows it to start. It is also used to simulate client and server CPI-C TPs in the same subarea as VTAM. TPs can have a conversation with any other TP on any LU to which VTAM allows a conversation to be started.

  This physical configuration contains WSim, VTAM, and VTAM applications, or TPs under test. WSim runs as a VTAM application program.

- TCP/IP application configuration

  This configuration is used to simulate Telnet 3270, 3270E, 5250, NVT, and FTP clients. These clients can have a session with any Telnet 3270, 3270E, 5250, NVT, or FTP server that TCP/IP allows. This configuration can also be used to simulate Simple TCP or UDP clients that are in session with various servers.

  This physical configuration includes WSim, TCP/IP, and TCP/IP applications under test. WSim runs as a TCP/IP application program.

Logical configurations

WSim can operate in one of the following basic logical configurations:

- VTAM application configuration

  This configuration is used to simulate SNA LUs that are accessing VTAM applications. LUs might be terminals or other VTAM applications.

  This logical configuration contains VTAM, VTAM applications, and VTAM applications and LUs simulated by WSim.

- CPI-C application configuration

  This configuration is used to simulate CPI-C client (allocates outbound conversations but does not accept inbound ones), TPs to test server (accepts inbound conversations), CPI-C TPs and network resources, or to simulate server CPI-C TPs to test client prototypes.

  This logical configuration contains VTAM, VTAM applications, and VTAM application CPI-C TPs and LUs simulated by WSim.

- TCP/IP application configuration

  This configuration is used to simulate TCP/IP clients in a TCP/IP network, or simple TCP or UDP clients that are accessing an application through a TCP/IP server.

  This logical configuration contains a TCP/IP server and any Telnet 3270, 3270E, 5250, NVT, FTP, simple TCP, and simple UDP clients simulated by WSim.

3.4.3 Script preparation

After the system configuration is defined, the definition of the network that is to be simulated must be defined next. This definition is done by creating a script, which contains the following parts:

- Network definition statements to describe the devices to be simulated by WSim
- Message generation decks to define messages to be sent by the simulated resources to the system under test

Network definition statements

Network definition statements specify the following information:

- Types of the simulated resources in the network
- Attributes of the simulated resources
- Connections between the simulated resources and the system under test
- Special information about delays, logic tests, the order in which message generation decks are used, logging, or tracing of the messages, and so on

NTWRK is always the first statement that is used to define a network. It names the network and specifies characteristics that apply to the network as a whole. It also can specify operands that establish defaults for lower-level statements. All other statements in the network definition follow the NTWRK statement in a prescribed order. The statements from the general simulation statements group immediately follow the NTWRK statement.

Different statements are used depending on what type of network is being simulated. For example, when simulating LUs that are accessing VTAM applications, the VTAMAPPL and LU statements must be used. When simulating CPI-C TPs, APPCLU and TP statements must be used. When simulating TCP-IP clients, TCP/IP and DEV statements must be used.

Not all statements are mandatory and some statements might be coded more than once. However, all of the statements in every group (including the optional groups) should follow the prescribed order.

Message generation decks

Message generation is the process by which terminals send and receive messages. Message generation decks are used to control messages that are being sent out and actions that are taken when messages are received by a simulated terminal.

A message generation deck contains one or more statements that are used to generate messages, set delays, define logic tests, define and control event actions, save data for future use, and so on.

Any terminal can use one or more message generation decks in any order.

The process to prepare message generation decks involves the following steps:

1. Decide what transactions to test.
2. Decide which application files and what data to use.
3. Create message generation decks by using one of the available methods.
4. Combine created message generation decks with network definition statements to form a script.
5. Test the script and modify and revise the script, if required.

In WSim, a *transaction* is an exchange of data between a simulated resource and the system under test. The choice of transactions depends on the objectives of the test. Usually, it is not necessary to test all possible transactions in the application. The following criteria can be used for the inclusion:

- Transactions taking the most processor time
- Transactions generating the most messages
- Transactions being the most important in the application

The following items also should be considered:

- The content of the messages to be sent
- The messages that are expected to be received
- The mix of transactions, such as the order in which WSim runs the message generation decks and which terminals use which decks
- The transaction rate

Use the PATH statement to specify the order in which the decks are run. Use the PATH operand on the DEV, LU, and TP statements to specify which paths a specified simulated resource runs.

Example 3-1 represents a fragment of the script for the RESNET1 network. The path SMALL specifies that the deck LOGON is run before the deck LOGOFF by the LU TERM1. The path LONG specifies that the decks LOGON, ALLOC, BROWSE, and LOGOFF are run in this order by the LU TERM2.

Example 3-1 Script fragments for the RESNET1 network

```
RESNET1 NTWRK
   .
   .
SMALL PATH LOGON,LOGOFF
LONG PATH LOGON,ALLOC,BROWSE,LOGOFF
   .
   .
TERM1 LU PATH=(SMALL)
TERM2 LU PATH=(LONG)
   .
   .
```

WSim executes the paths repeatedly; that is, when the terminal has executed the last deck in the path that is defined for it, it starts again with the first deck in its path. Terminals maintain their positions in the paths and are not affected by other terminals. BRANCH, CALL, and IF statements can be used to alter linear sequences of paths. The order in which WSim executes decks in any path can be certain, random, or based on probability distribution.

WSim can generate messages with controlled intermessage delays. This configuration can be used to simulate the delays of real operators as they view the window, think about the information, or enter more data. Intermessage delays can be defined for the entire network, a specific resource, or even on a message-by-message basis.

Before you create message generation decks, the transactions to be tested should be thoroughly analyzed. All steps should be listed.

Methods for creating message decks

WSim provides the following methods for creating message generation decks:

- Directly written message generation statements
- Programs in Structured Translator Language (STL)
- Use of one of the script-generating utilities that are provided with WSim to convert captured data traces

WSim provides the following script-generating utilities:

- Interactive Data Capture (can produce STL programs)
- Script Generator utility
- SNA 3270 Reformatter Utility

The method that used depends on what is being tested and on the following factors:

- Familiarity with WSim
- What kinds of messages are sent to the system under test by WSim

It makes sense to trace actual system activity and use the Script Generator utility to convert the trace records if the test involves simulating a number of real users who are using an application.

Some or all of the methods might be used when real tests are prepared.

Writing message generation statements

Knowledge of message generation statements is important when the output from the STL translator is interpreted, when the script generation utilities are used, and when the scripts are debugged.

The message generation statements must be coded manually in the following situations:

- When the output from one of the script generation utilities is modified
- When more message generation decks in a script that is produced by the STL Translator or one of the script generation utilities are added
- When some special types of messages or special conditions in an SNA network are added
- When existing message generation decks are modified

The syntax for message generation statements is similar to that for network definition statements.

Use the preprocessor to check the syntax and store message generation statements in data sets for use in simulations.

Using STL and the STL Translator

STL is a high-level, structured programming language that can be used to create message generation decks and define terminals and devices to be simulated by WSim. STL uses constants, variables, expressions, and structured control statements.

An STL program is usually divided into one or more procedures. The STL Translator translates STL programs into message generation decks. Each message generation deck corresponds to one STL procedure. Network definitions can be included in STL programs.

The STL Translator invokes the preprocessor to validate and store the network definition statements.

Example 3-2 shows two simple STL procedures. Procedures begin with an MSGTXT statement and end with an ENDTXT statement.

Example 3-2 Message generation decks written in STL

```
/* STL procedure logging terminal on to RESAPPL */
Logon: Msgtxt
Initself('RESAPPL')
Endtxt
/* STL procedure testing message generation */
Tstmsg: Msgtxt
Do i = 1 to 5
Type "Hello, I expect you to respond Hi"
Transmit using PF4,
and Wait until on substr(screen,40,2) = "Hi"
End
Endtxt
```

The first STL procedure, LOGON, defines the text that a terminal uses to log on to an application named RESAPPL. When the second STL procedure, TSTMSG, is executed, WSim simulates a user who is typing `Hello, I expect you to respond Hi` and then pressing PF4 to send the message to the application. WSim waits for the application response `Hi` to appear at position 40 on the screen. These messages are sent five times.

The STL Translator can be started by using JCL, a TSO CLIST, or the WSim/ISPF interface.

Using the Interactive Data Capture Utility

The Interactive Data Capture Utility (IDC) ITPIDC is a host application that can capture 3270 device session data and generate scripts. A user logs on the same way as for any other VTAM application, and through it can log on to the VTAM application to be tested and perform all of the actions to be simulated by WSim. IDC that is capturing the session traffic is not apparent to the VTAM application.

From the captured session data, IDC can directly generate an STL program, WSim message generation decks, or both.

Using the script generator utility

The script generator utility creates message generation decks that are based on traces of real users who use real applications. The captured trace must be put in a specified format and sorted by resource name, date, and time. The sorted trace is used as an input for ITPSGEN, which generates the message generation decks.

The following methods can be used to obtain a system activity trace:

- The NetView® Performance Monitor (NPM) capturing path information units for selected LUs
- The Generalized Trace Facility (GTF) capturing the VTAM Buffer Trace
- User-written capture routines

WSim provides a special program ITPVTBRF to help with reformatting traces, which are not in the format that is required by ITPSGEN.

ITPSGEN also requires complete, syntactically correct network definitions as input. It uses the network definition statements to determine the terminal names for which to generate the decks. The names in the DEV and LU statements must correspond to the resource names that are used in the trace.

Using the SNA 3270 Reformatter Utility

The SNA 3270 Reformatter Utility (ITPLU2RF) is a batch utility for reformatting NPM log records (FNMVLOG) from LU2 sessions into log records. ITPLSGEN can be used to create STL programs or message generation decks that are based on ITPLU2RF output.

Testing scripts

The scripts must be tested to ensure that they are coded correctly, and that they function as intended. Statement syntax can be checked by using the Preprocessor or the STL Translator. The following methods can be used to ensure that the message generation decks function as intended:

- Message trace records tracing the message generation process
- STL trace records tracing the message generation process for STL programs
- Self-checking scripts

To ensure that unexpected situations encountered during simulations are handled properly, use self-checking scripts. They do not have to be used for all simulations (for example, they can be skipped for short and simple simulations). However, they must be considered for a long-running test, which could be wasted if terminals were to go out of synchronization.

IF statement logic tests are added to the scripts to check for the expected response and to take action if an unexpected one is received. This action could be simple (such as stopping the device) or complex. The action could include several possible courses of action that are based on the actual response. The logic tests can be written in decks that are created by STL, one of the script generation utilities, or manually. They can also be coded in network definitions.

3.4.4 WSim output

WSim provides several online and printed reports to analyze test results. Some reports are produced by default, but some must be requested by issuing specific operator commands or running one of the WSim utilities. The following types of reports are available:

- Operator reports that indicate what is happening during operation.
- The complete message log.
- Reports that are generated by the following utilities that are based on the message log:
  - Formatted reports that are produced by the Loglist utility.
  - Reports on the differences between 3270 display records in two message logs that are produced by the log compare utility
  - Detailed statistical analysis of response times that are produced by the response time utility.
- Online response time statistics.

Most of the reports are intended to represent how WSim is interacting with the system under test and not the effectiveness of the network or the application.

Interval reports monitor the current activity and status of each simulated resource in the network. The statistics are accumulated until the network is canceled or reset. End-of-run reports provide summary data from the simulated network. They are produced automatically and have the same format as interval reports. The inactivity report contains information about each inactive resource in the network.

The log data set is the single most valuable tool for debugging the scripts. This data set contains all data that was transmitted or received by the WSim simulated resources. The message logging facility is active for the entire network, but it can be deactivated completely or partially for a VTAMAPPL statement in the network. A separate log data set can be used for a particular network. This feature is convenient when multiple networks are run because the results are logged separately.

The Loglist utility uses the log data set. The control commands can be contained in a file or entered at the operator console. You use the WSim/ISPF interface, JCL, or TSO CLIST to start the utility, name the input files, and specify where the formatted log is printed.

The Loglist utility uses different formats for each type of log records. One useful feature is the ability to print screen image records. These images are updated each time that a message is sent or received by the device. The output from the Loglist utility for this type of log records looks the same as the screen images a user sees at the real device.

3.4.5 Operating WSim

WSim can be run by using JCL, TSO CLIST, or the WSim/ISPF Interface.

The sample JCL can be found in the WSIMPRC6 member of the data set HLQ.SITPSAMP. The sample TSO CLIST can be found in the member WSIMRUN of the data set HLQ.SITPCLS. The value of HLQ and the method to invoke the WSim/ISPF Interface depend on how WSim and this interface are installed on the site. A typical WSim/ISPF main panel is shown in Figure 3-10.

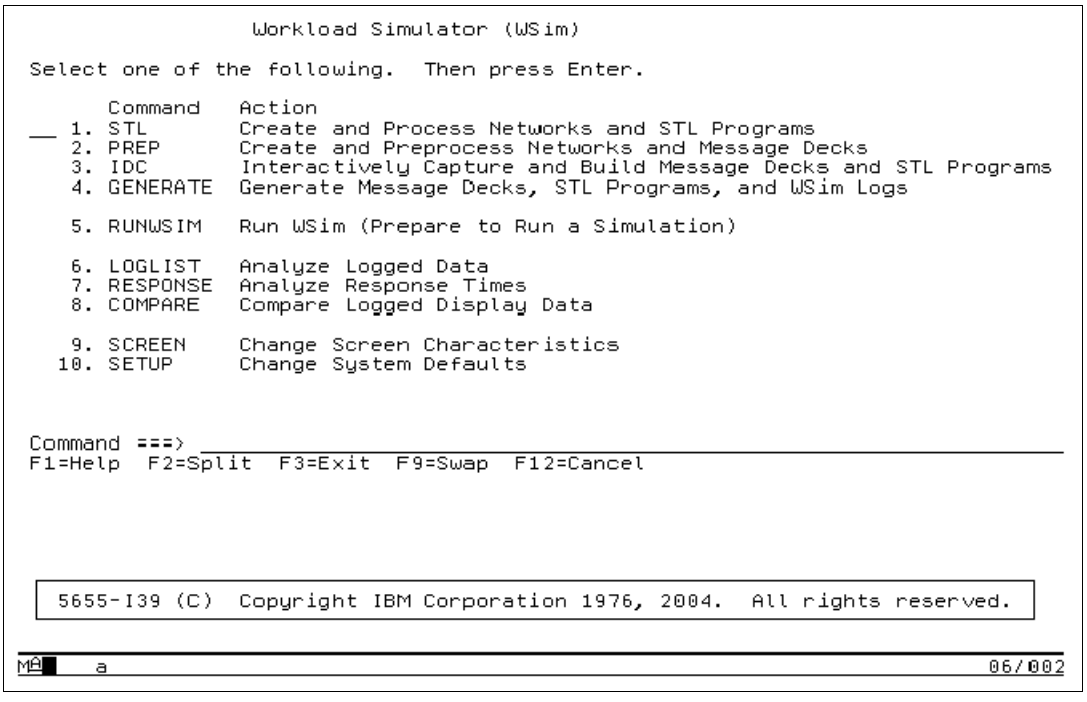


Figure 3-10 WSim/ISPF main panel

3.4.6 WSim Test Manager

The WSim Test Manager (WTM) is a usability enhancement that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation.

The primary concept of the WTM testing structure is a project, which is a set of libraries that contain schedules and test scenarios. Projects can be archived and reused. A project must be created before any schedules or test scenarios can be created by using WTM.

A WTM schedule is a WSim network definition and the associated test scenario definition.

Test scenarios are organized into the following levels:

- Test case: Can be reused within multiple test groups
- Test group: An ordered list of test cases
- Test cycle. An ordered group of test groups and test cases

WTM offers various ways to automate the development of test cases, which are WSim scripts that are written in STL. For 3270 environments, WTM can automate the script generation process from 3270 screen and keyboard captures (IDC), SNA traces, WSim or IDC logs, or from STL models and skeletons. Automated CPI-C test case generation uses SNA traces. The STL source is automatically translated into WSim MSGTXTs.

Generated test cases are paired with network resource definitions as part of developing WTM schedules. The WTM schedule is used by WTM to define and control the WSim simulation run (test). WTM schedules can be archived and reused.

The typical WTM main panel is shown in Figure 3-11.

```
                          WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                      Alternate HLI:

Licensed Materials - Property of IBM.
5655-I39 (C) Copyright IBM Corporation 1993, 2004. All rights reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.

Command ===> ____________________________________________________________




 F1=Help      F2=Split     F3=End       F4=          F5=          F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          06/002
```

Figure 3-11 Typical WTM main panel

3.4.7 Latest enhancements

Applying the PTF, which fixes the APAR PQ94132 for the Workload Simulator, provides several general enhancements to this tool. The following enhancements are of high significance:

- Password masking on formatted 3270 screens
- WSim Adapters for Rational TestManager

Passwords are usually maintained on the 3270 screens in unprotected non-display fields. Although they are not visible, the passwords are sent to host applications in the clear and so are captured by the Interactive Data Capture utility or generated by script generation utilities.

The enhancement masks passwords by encrypting or hiding them by using asterisks in test scripts and logs. The utility ITPGNKYZ is supplied to generate required USERMODE.

The Workload Simulator Adapters for Rational TestManager allow WTM schedules and JCL scripts to be started from the IBM Rational TestManager that is running on a remote workstation.

To run WTM schedules from Rational TestManager, the schedules must exist in WTM on the host system. Some migration steps must be performed first. Also, a user ID and a password for a TSO user and the user ID of the WSim user (who created WTM projects and schedules) are required to run the WTM schedules from the Rational TestManager.

The white paper *IBM Workload Simulator Adapters for Rational TestManager Version 1, Release 1.0.1* and the installation program are included with the PTF.

4

Using the IBM Problem Determination Tools Graphical User Interfaces

This chapter provides an overview of the graphical user interfaces (GUIs) that are available for interacting with the IBM Problem Determination Tools. We assume that the current version of plug-ins (V12) was downloaded and installed. The illustrations that are provided were created with the Problem Determination Tools GUIs installed into CICS Explorer. However, their function remains the same in any Eclipse-based platform. The Rational Developer for System z provides additional integration that also is described in this chapter.

4.1 Problem Determination Tools plug-ins: Overview

By using the supplied Eclipse plug-ins, the IBM Problem Determination Tools can be driven through a GUI. Each plug-in must be installed into an Eclipse-based platform, such as IBM CICS Explorer or Rational Developer for System z. The complementary server support must be configured on the host.

The PD Tools GUIs provide a workstation alternative to the z/OS ISPF interface. With this new interface, existing application developers and system programmers can experience productivity gains over the traditional green screen interface. The new interface also can reduce the learning curve for new z/OS developers. Reducing the learning curve increases proficiency more quickly by providing an Eclipse-based interface to the z/OS tools that is easier and more familiar for new hires to learn to use.

4.2 Application Performance Analyzer

The Application Performance Analyzer (APA) plug-in enables application developers and systems programmers to analyze the performance of z/OS applications by using Application Performance Analyzer for z/OS with a GUI.

4.2.1 Introduction

The APA GUI can be used for submitting new observation requests and for browsing the performance analysis reports that are generated from observation requests.

Host requirements

For the V11.1.0.2 plug-in, use Application Performance Analyzer for z/OS V11.1, with PTF UK67481(Base) for APAR PM34080, or later.

The graphical user interface

After the APA GUI is installed into the CICS Explorer, you can begin creating observations and viewing observation reports.

4.2.2 Opening the APA plug-in perspective

Complete the following steps to access the APA for z/OS GUI:

1. Click **Window** → **Open perspective** → **Other**, as shown in Figure 4-1.

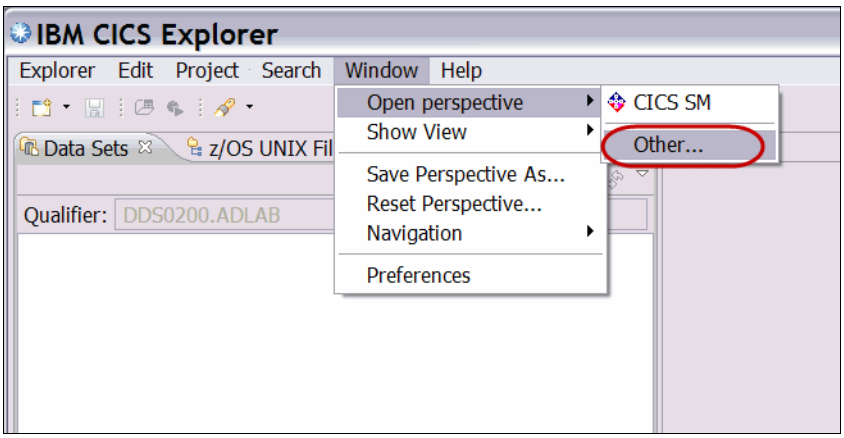


Figure 4-1 Open perspective

2. Select **APA/GUI** and click **OK**, as shown in Figure 4-2.

Figure 4-2 Select APA/GUI

3. If a Logon window is not shown, click the arrow button in the upper left corner of the IBM CICS Explorer window, as shown in Figure 4-3.

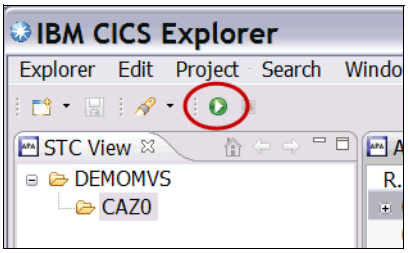


Figure 4-3 Connect APA

4. When the logon window is shown, enter the Host Address, Host Port, and your TSO user ID and password to connect to z/OS. Contact your system administrator for this information if you do not have it.
5. Open the File Manager plug-in view, as shown in Figure 4-4.

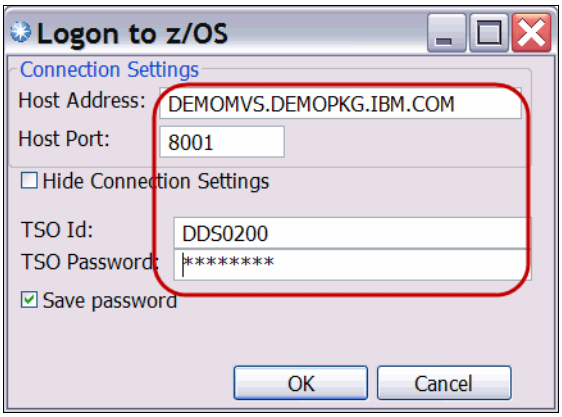


Figure 4-4 Log on to APA on z/OS

The fields that you must complete can include the following values:

- Host Address: The z/OS server where the Application Performance Analyzer for z/OS GUI communications (listener) is installed. This address is the host address of the machine.
- Host Port: The port where the z/OS Application Performance Analyzer server is listening. This port is the port number that the listener is configured to use.
- TSO ID: Your logon TSO ID.
- TSO Password: The password for your TSO ID.

The address, port, and TSO ID are saved as a Network Connection Preference the first time they are entered. If any of these values are changed in a subsequent login, they are not saved as a preference. Instead, they are kept only for the connection session.

The Network Connection Preferences window includes the settings to change and save these preferences.

6. Access the Preferences page by selecting **Window** → **Preferences** → **APA/GUI** → **Network Connections**, as shown in Figure 4-5.

   The Network Connections preferences are the default connection parameters that are used in the logon dialog. You can select the default APA STC (Started Task) from the pull-down menu that is called Other. This menu includes a list of active APA STCs, or you can manually enter it.

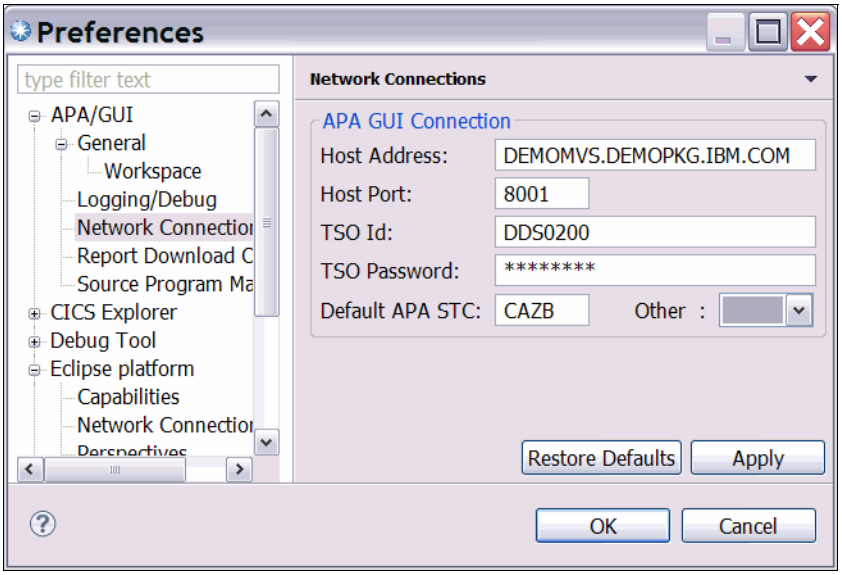


Figure 4-5 Network Connections preferences

7. Click **Apply** to save the changes to your workspace. The changes can be used for future connection attempts.

After a successful connection to z/OS is established, the Application Performance Analyzer for z/OS GUI automatically populates with the Observations List of the remote repository Default APA STC, as shown in Figure 4-6.

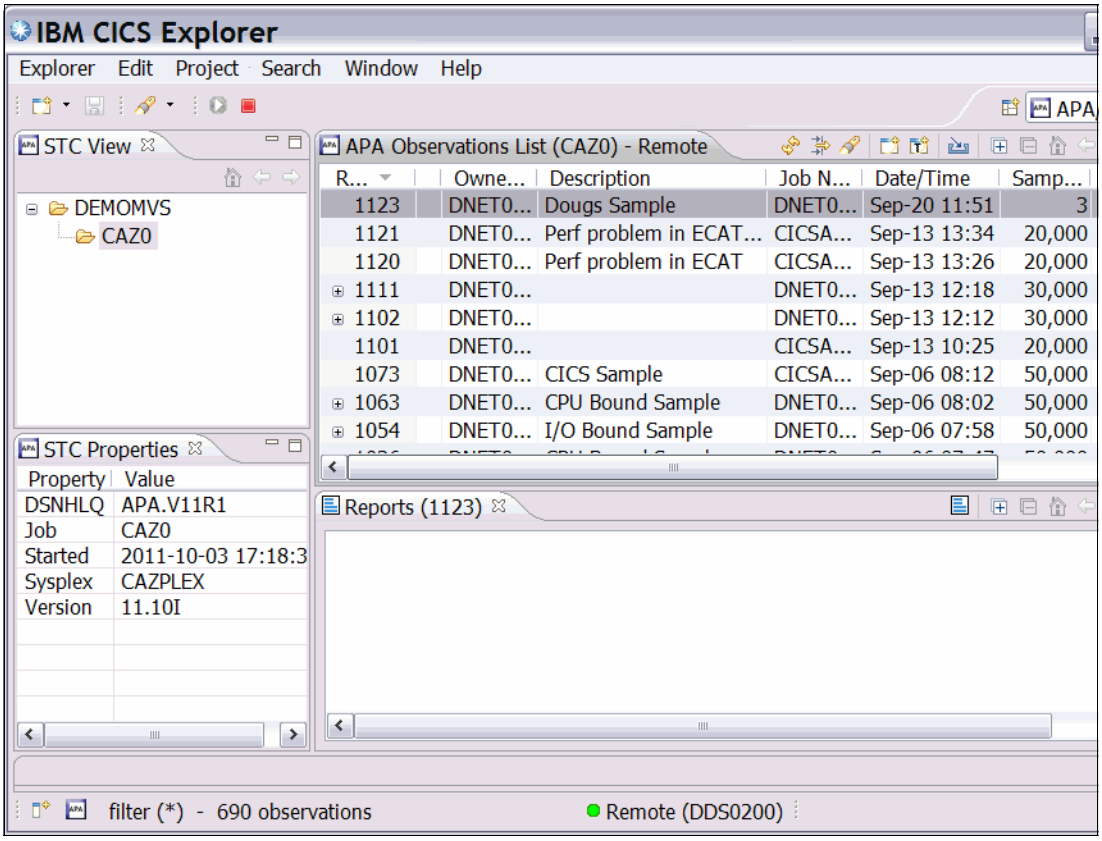


Figure 4-6 APA connected to host

4.2.3 Working with the APA perspective

The APA perspective is a collection of menus and views that are displayed and provide functions to multiple components of APA simultaneously. The following major views are included:

- STC View, which lists all active started tasks
- Observations List View, which lists all observations
- Observation Detail View, which provides details of an observation
- Reports List View, which lists all reports for an observation
- Report View, which displays an individual report

Main menu action bar

At the upper left of the APA GUI is the Main menu bar. By using the Explorer pull-down menu, you can connect to, disconnect from, and exit APA. The Search menu controls the search actions. By using the Window pull-down menu, you can select the views within the perspective and work with your preferences. The Help menu gives access to help topics for the APA GUI, as shown in Figure 4-7.

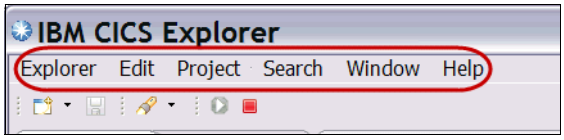


Figure 4-7 Main menu action bar

Status line

At the bottom of the APA GUI is the status line, as shown in Figure 4-8. The far left icon is the FastView button, with which a view can be minimized to an icon in the status bar. You can click this icon at any time to open or minimize the view.

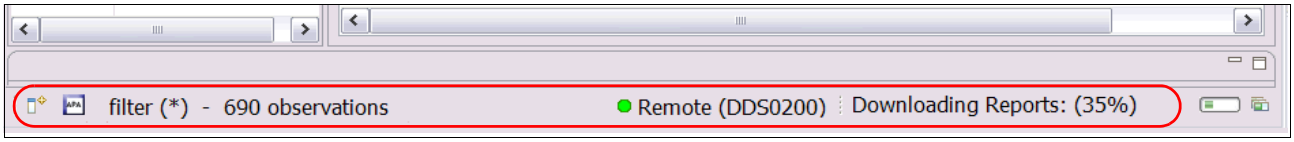


Figure 4-8 Status line

The Observation view status information is next, which gives details about the Observations in the Observations View, the filter that is in use (“ * ” is a wildcard), and the number of displayed observations. The application status information is the indicator for connection to the host.

Finally, the Job Progress information informs you about the status of long running tasks.

STC view

The STC view lists all of the APA tasks that are running on the sysplex to which the APA GUI is connected. By using this view, you can switch between the started tasks to view separate observations and observation lists. The properties for the selected started task are listed in the STC Properties view, as shown in Figure 4-9.

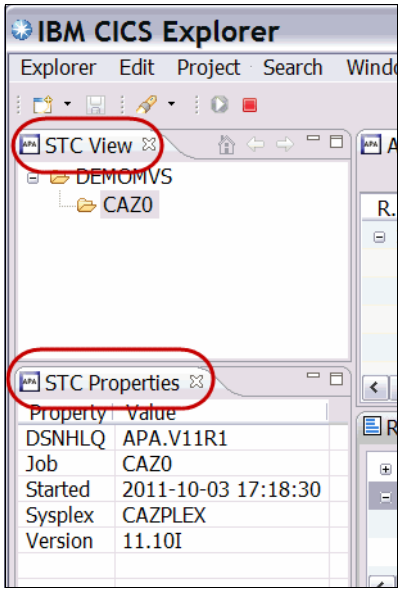


Figure 4-9 STC views

Observations list view

As shown in Figure 4-10, the observation list view is where a majority of the functionality of the APA GUI is shown. From this view, you can view a list of all of the observations that were submitted for the started task, and sort them based on columns or user-defined criteria. You also can use the menu to perform a variety of observation-related tasks.

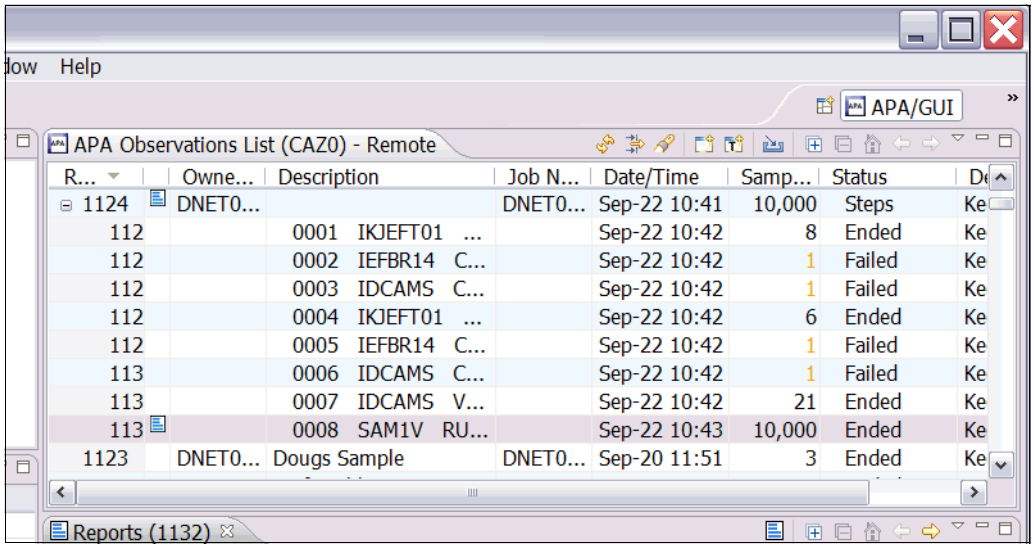


Figure 4-10 Observations List view

Each observation is saved on an individual row. After an observation is complete, the rows can be expanded to show the various parts of the observation. Reports that were downloaded from the host to the APA GUI are indicated with the Reports icon.

Reports list view

As shown in Figure 4-11, the observation reports list view displays a listing of all the reports for the selected request in the observations view. The observation report is downloaded to get the list. The reports are arranged in a two-level tree structure. The first level is the report category, and the second level is the individual reports.

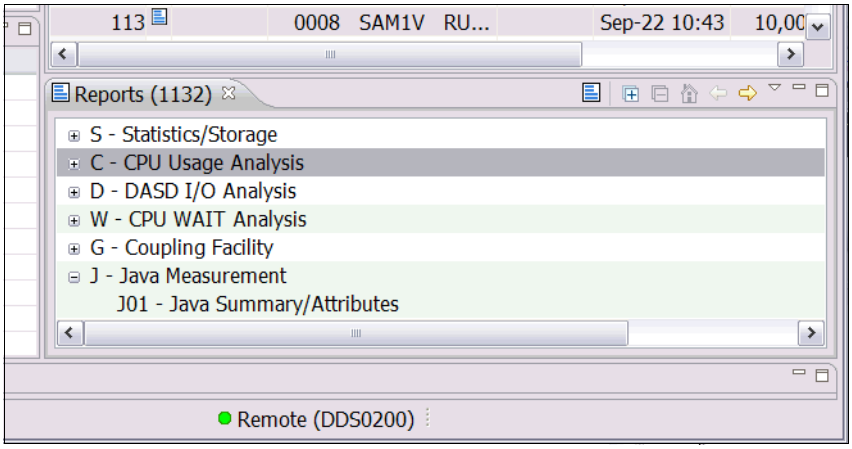


Figure 4-11 Reports list view

Reports view

The reports that are selected from the observation report list view are displayed in the reports view, as shown in Figure 4-12. After a report is selected, the report is shown in the report view. Multiple reports can be opened and viewed, including reports from observations that are not currently selected.

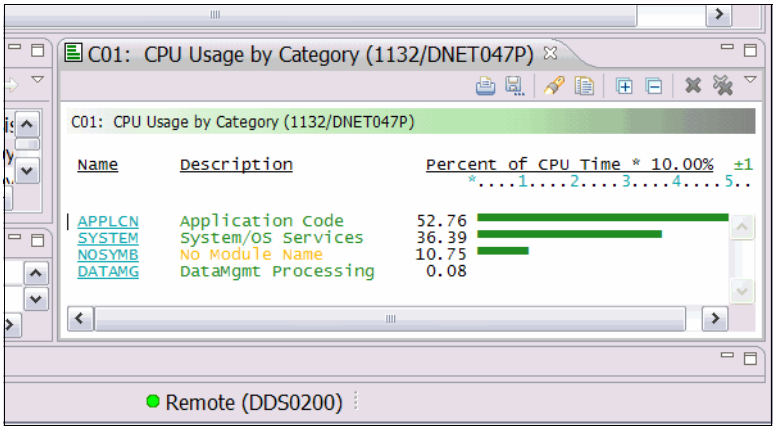


Figure 4-12 Single report view

4.2.4 Submitting an observation request

New observation requests can be created and submitted by using the APA GUI. Creating a request can be done from the Observations List view. There are various ways to begin the observation request dialog. In the following example, we show a method that uses the New Observation button.

1. Click **New Observation** to schedule a new measurement, as shown in Figure 4-13.

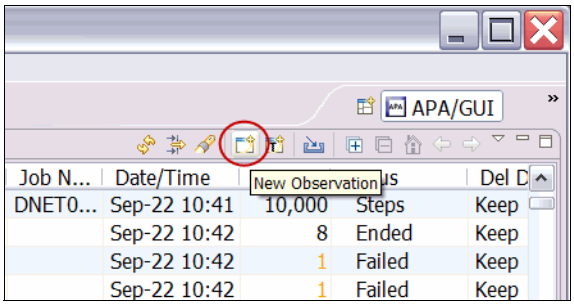


Figure 4-13 New observation

The New Observation window opens. This window is used to specify the parameters for the observation request. You are not required to use all of the tabs to set up the observation.

Any tabs with entered information feature a red or green dot on the tab. A green dot means that the required information was entered to submit the observation request. A red dot indicates that the information includes errors or that more information is needed.

If any of the tabs include a red dot, you cannot submit the new observation, as shown in Figure 4-14.

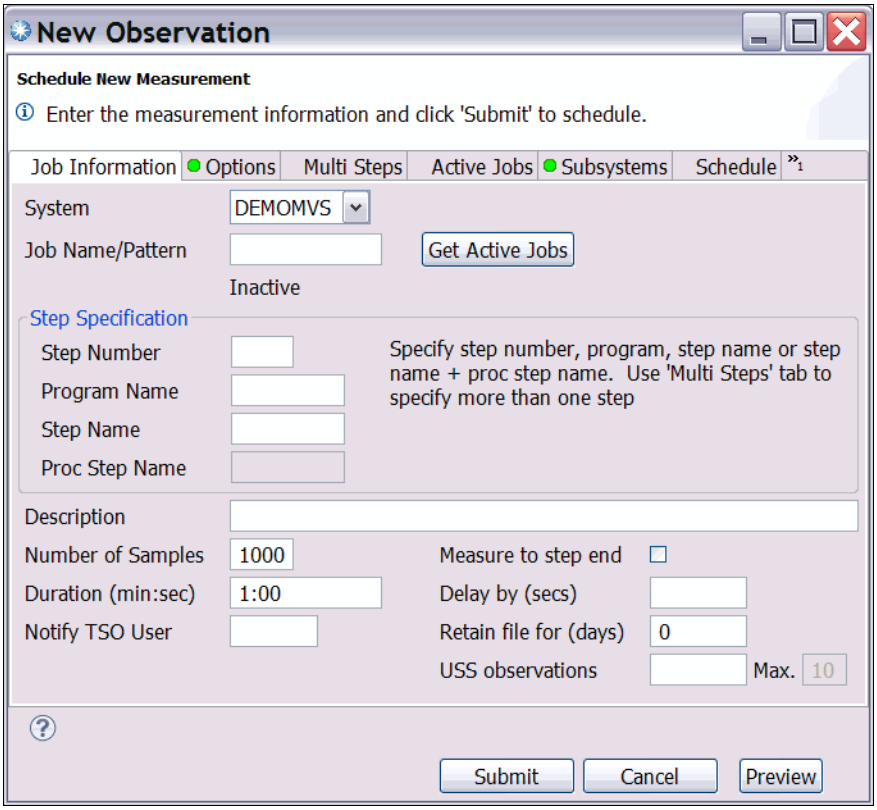


Figure 4-14 New Observation window

There are seven tabs in which you can enter information for a request. Each panel is listed as a separate tab at the top of the window. For example, as shown in Figure 4-14, the first tab is called Job Information. The following tabs are available from the New Observation window:

- Job Information tab

  When you enter a new observation request, a job name is required. Enter the name of the job, region, or started task that you want to monitor in the **Job Name/Pattern** field and click **Get Active Jobs**.

  You also can enter a partial job name that includes a wildcard. For example, if you enter **ABC*** and click **Get Active Jobs**, you can obtain a list of running jobs that start with ABC. Enter the * wildcard into the Job Name/Pattern field and click **Get Active Jobs** to display a list of all jobs. For online applications, enter the name of the CICS or IMS region that you want to monitor.

Also included in the panel is the Step Specification area. By default, APA monitors only one step. For an active job, it monitors the step that is running at the time that you enter the request. If you schedule a request for a job that will run in the future, the default action is to monitor only the first step. To remove the default action, leave all of the fields in the Step Specification area blank. The Step Specification area includes the following fields:

- Step Number: Enter the * wildcard in this field to monitor every step in the job, or enter a number to monitor any individual step number.
- Program Name: Enter a program name in this field.
- Step Name: Enter a step name or a combination of the step name and procstep name. (There are more options for monitoring multiple steps in the same job on the Multi Steps" tab, which is described later in this section.)

The Job Information tab also includes the following fields:

- Description: Enter a description of the observation.
- Number of Samples: Specify the total number of samples that APA must snapshot during its monitoring session.
- Duration: Specify the measurement duration in minutes and seconds (min:sec). For example, enter the number 1:00 to monitor the application for 60 seconds. Enter 2:15 to monitor for 2 minutes and 15 seconds, or 2: to monitor for 2 minutes. For a typical application, you need APA to monitor only the application for a few minutes. However, there might be situations in which you want to continue monitoring until the job or step ends. To do set this parameter, click Measure to step end.
- Notify TSO User: Leave this field blank if you do not want to be notified when APA completes its observation. You can enter another user's ID so that they are notified instead.
- Delay by (secs): Enter a delay time (in seconds) that elapses before APA begins taking samples.
- Retain for (days): Specify the number of days after which APA automatically deletes the measurement data sets it used to collect its monitoring information. Even if you do not specify anything here, measurement data sets are deleted automatically if you delete the observation from the list.

- Options tab

  In this tab, you set the parameters for APA to collect detailed information about CICS, DB2, IMS, Java, or IBM MQSeries®. APA collects information only about the subsystems that are selected in this panel. For example, if you monitor a CICS application that accesses DB2 databases, by default you cannot capture any details about CICS transactions or SQL statements. You must specify in this panel any of the data extractors that you need.

  There are several levels of data extractors available for DB2 and IMS. If your application uses DB2 or IMS, you must enable all of the extractors for DB2 or IMS. By doing so, you are asking APA to do a little more work, but you are rewarded with much more in-depth information about your application. You might be tempted to enable all of the extractors for every application. In practice, you might not notice an excessive amount of extra monitoring overhead. However, is best to request only the extractors that your application really needs.

- The Multi Steps tab

  In this panel, you can specify a single step to monitor, or you can request that all steps are monitored by entering the * wildcard. By using this tab, you have more control over which steps are monitored. For example, if you can monitor the 4th, 5th, and 10th steps of a job by setting such parameters here. On each individual line, you can enter a step number, program name, step name, or a combination of step name and procstep name.

- Active Jobs tab

  This panel displays a list of jobs that are running, and you can select a job in the list to monitor. To get a list of jobs, enter a job name or the first few letters and a * wildcard in the prefix field. You also can leave the field blank to get a complete list. Click **Get active Jobs**.

  You can select a job to monitor by a selecting the box to the left of the job name. After the job is clicked, it is added to the Selected Jobs List. To remove a job from the selected list, clear the box.

- Subsystems tab

  The subsystems tab gives access to CICS, DB2, and IMS subsystem information. The default subsystem that is shown is CICS. You use this panel only when you are monitoring CICS applications.

  You can use the panel to limit which CICS transactions and which CICS terminals are monitored. If the CICS data extractor is on, then by default all of the CICS transaction codes and all of the CICS terminals are monitored. If you are working with an application that is running in a CICS region with a high transaction volume, you might find it useful to monitor only specific transactions and terminals. However, many users prefer to always monitor all transactions and terminals. Even when all data is collected, it easy to look at specific transactions when you are viewing APA reports.

  Click the IMS radio button to enter IMS transaction information. Entering a dash in the Job Name field in the Job Information tab enables the DB2 radio button and you can enter the DB2 Subsystem, schema, and program or function name for the DB2 stored procedure or function that you want to measure.

- Schedule tab

  Use this panel to schedule observation sessions that you want to add. Click **Calendar** to show the calendar and select the date and time. In the Measurement repetitions area, you can select the number of times the observation is to repeat, and the frequency for the repetitions. Finally, click **Add to Schedule** to add the date and times to the upcoming observation schedule. The observations remain in the Pending ADD status until the observation is submitted.

- Sched Options tab

  In this panel, you specify the option that controls when APA starts its monitoring sessions. It is important that you understand this option because it controls when APA monitors an application and when it does not. Setting this option incorrectly can prevent APA from monitoring your application. The setting for this option depends on whether the job you want to monitor is already running, or if it is not running but is going to start later.

  In the Measure active job field, select box if the job you want to monitor is already running. When you enter the observation request, APA immediately begins its monitoring session on the job. If you leave this field blank, you are telling APA to start monitoring the job the next time that it starts. Be careful about this setting because if you specify N (for No), APA does not monitor your job if the job is running. Instead, APA waits until the next time a job by the same name starts.

  This tab also is used to specify the action to take if the job runs more than once within a specified interval.

Submit the observation request

After you enter all of the required parameters into the observation request, you can submit the request. At any time as you build the observation session, you can click **Preview** to view the text version of the observation session settings. This preview shows the details of all of the specifications that you entered, as shown in Figure 4-15.

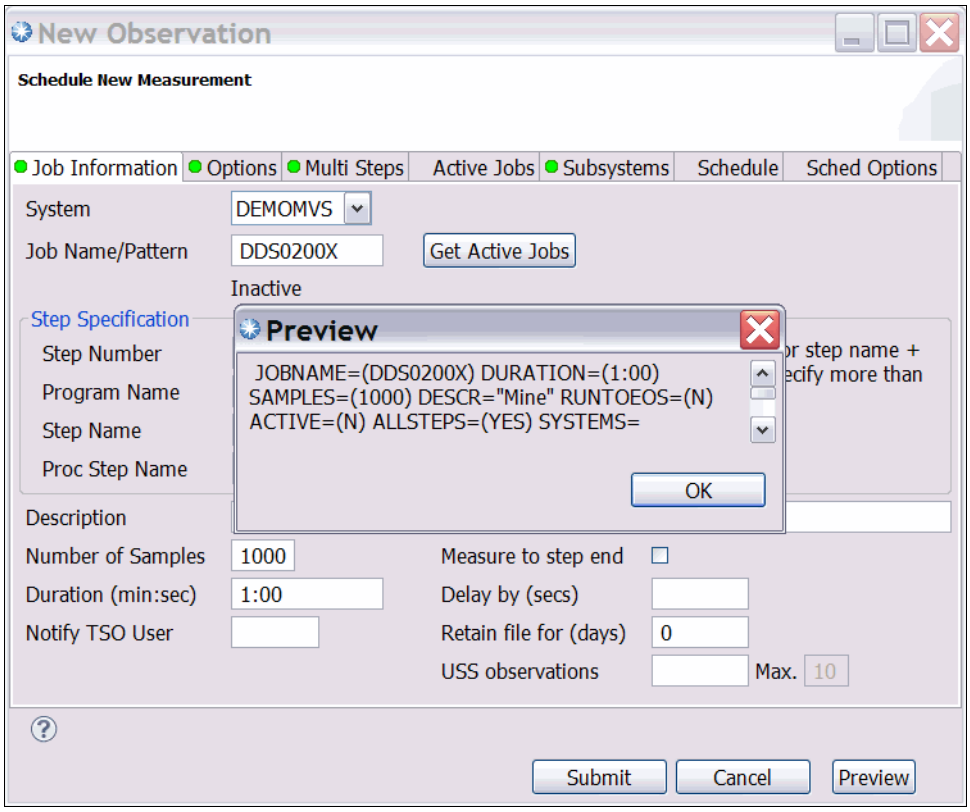


Figure 4-15 Observation preview

Click **Submit** to send the request to the host system, and a new observation is shown in the observation list, as shown in Figure 4-16.

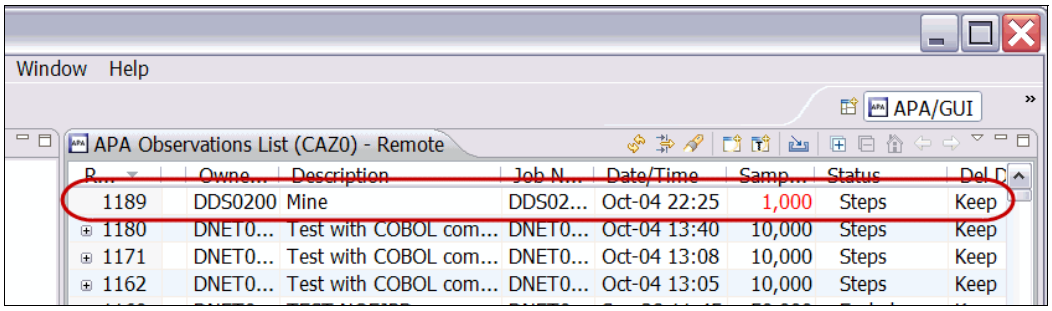


Figure 4-16 Added observation request

Threshold observations

Threshold observations are observations that are not triggered to begin until certain parameters for an application such as, wait time, CPU time, elapsed time, or exception count is exceeded. Click the icon for Threshold Observation on the Observation List action bar to begin creating an observation, as shown in Figure 4-17.

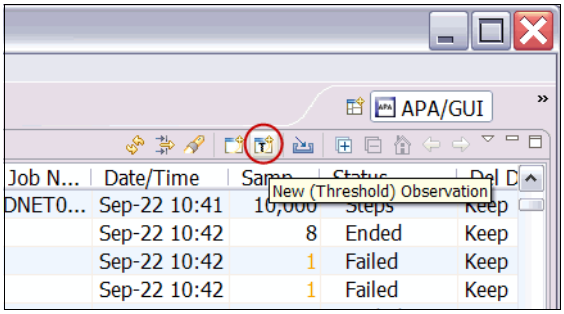


Figure 4-17 New threshold observation

The basic information that is needed for the new observation is the same information as the standard observation. However, the parameters to trigger the observation must be added. As shown in Figure 4-18, in the Criteria tab, enter the criteria that triggers the observation. All criteria that is entered must be met before the observation begins.

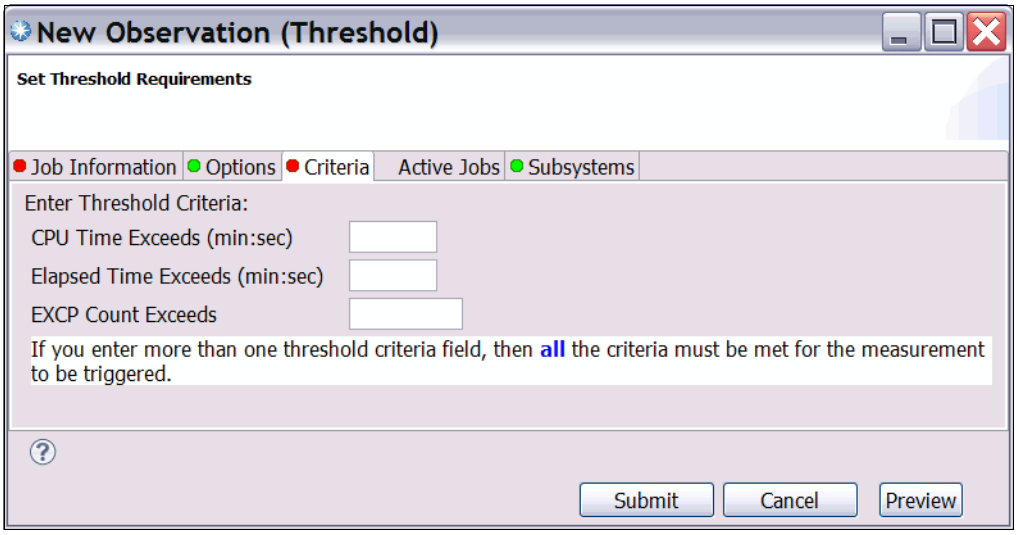


Figure 4-18 Threshold observation criteria

4.2.5 Downloading and viewing a report

After the observation runs to completion, you can download the generated reports and view them by using the APA GUI.

The main portion of the standard setup is the Observation List view, which displays a list of the completed and scheduled observation requests. Right-click in the view to open a context menu and choose additional actions to perform.

Locating the observation

Finding a particular observation in the observation list is made easy by clicking **Filter Observations**, as shown in Figure 4-19.

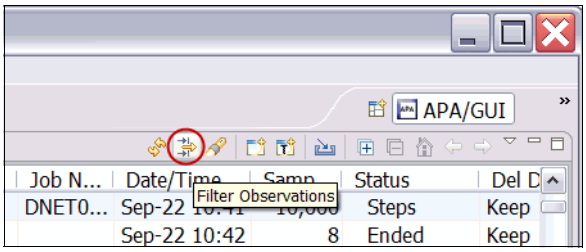


Figure 4-19 Filter Observations icon

By using the Filter, you can narrow down the list of observations that use UserID, Job Name, and sort sequence, as shown in Figure 4-20.

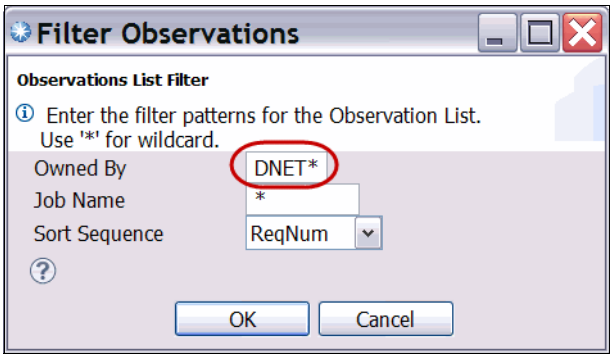


Figure 4-20 Filter Observations window

The result is only those observations that met the filter criteria are shown in the Observation List, as shown in Figure 4-21.

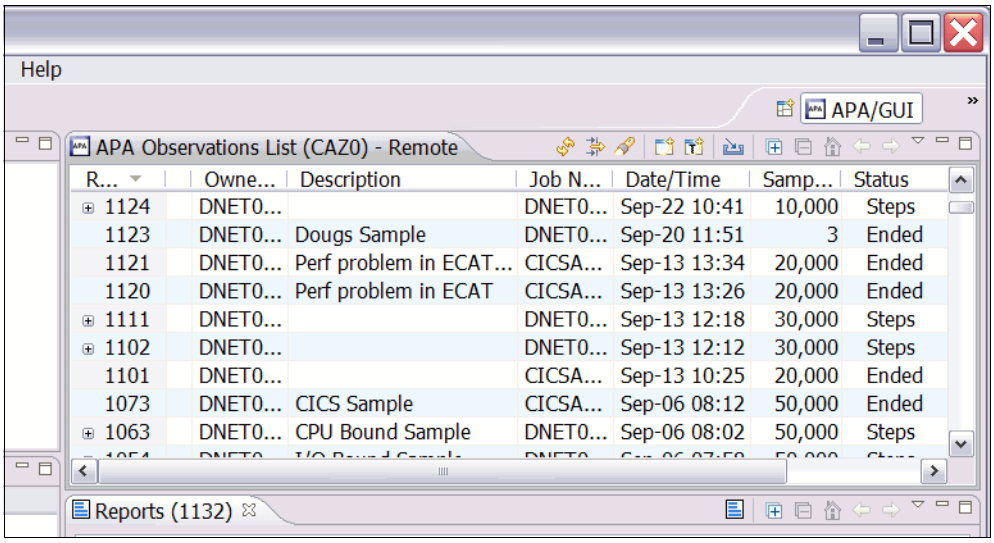


Figure 4-21 Filtered observation list

Downloading the observation reports

When the job started to run, APA started to monitor it. Because APA is monitoring multiple steps for this job, you can expand the entry to display each step that is monitored. To expand an entry, click the small circle X that is in front of the ReqNum, as shown in Figure 4-22.

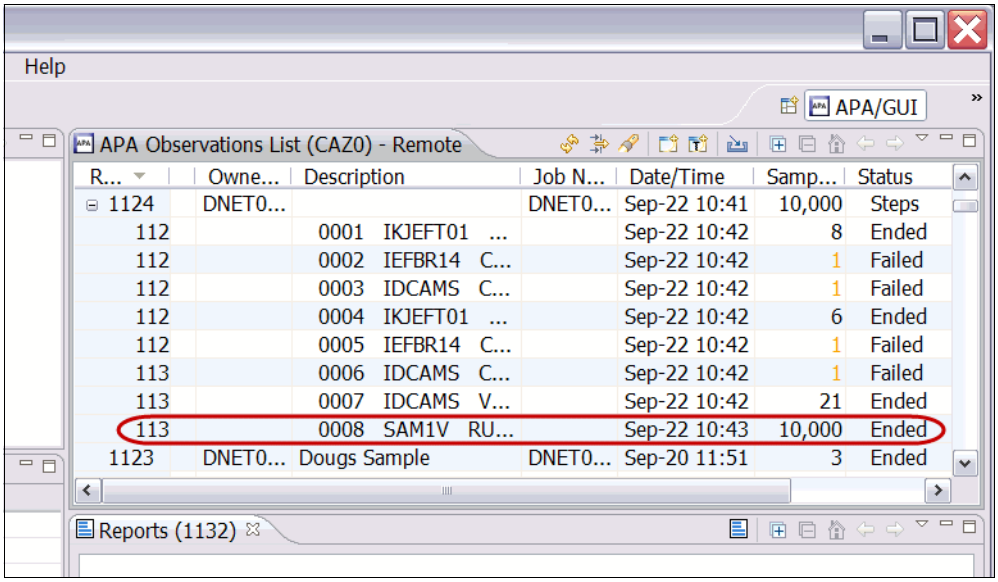


Figure 4-22 Expanded observation

Any of the observation steps can be downloaded. For this example, we review the SAM1V step observation reports. Right-click the observation step that you want to download and select **Download Reports**, as shown in Figure 4-23.

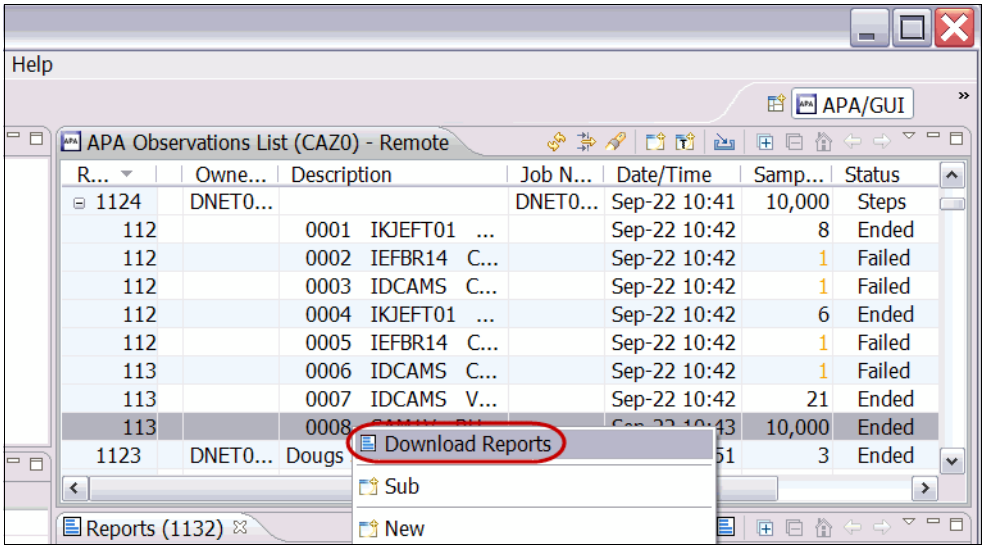


Figure 4-23 Download reports

The APA GUI contacts the host and downloads a copy of the report to your workstation. The Downloading Reports window shows you the status of the download. After the download is complete, a small Report icon is shown next to the observation step, as shown in Figure 4-24.

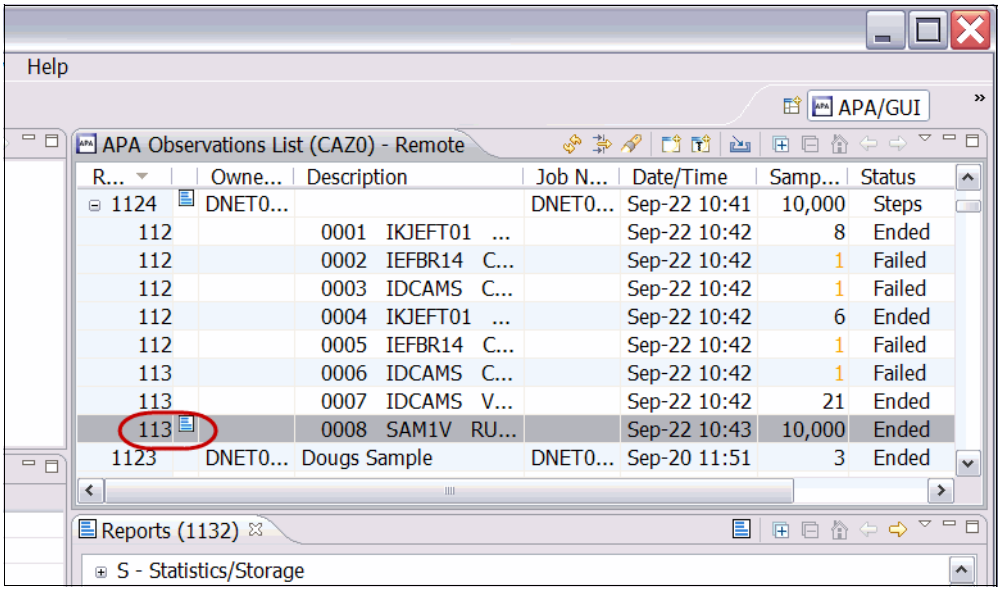


Figure 4-24 Reports Icon

Viewing reports

After the reports are downloaded, you can select a report from the Reports List View. The reports view shows the performance reports for this entry. Notice that the view contains report categories. Each category can be expanded to show all of the reports that are contained within them by clicking the "+" in front of the report category.

After a report category is expanded, clicking a subcategory opens the report for viewing. In this example, the CPU Usage Analysis report category is expanded, and selections for those reports are shown. In the Reports view, the C02 - CPU Usage by Module report is selected and opened, as shown in Figure 4-25.

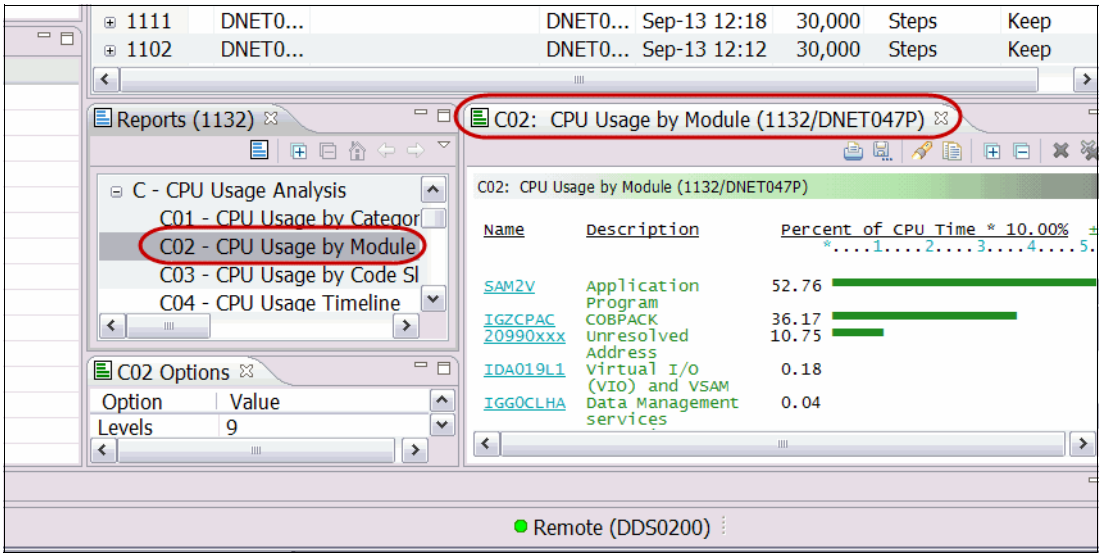


Figure 4-25 Reports view

You can open additional reports at the same time by clicking their respective report names.

Source code mapping

Source program mapping displays the source code for a requested report line. This option is available only for all report lines that are related to application programs. It also requires the setup of a source mapping repository that provides a list of libraries and directories to locate the source code. When selected, a new window shows the source code information contents. The nature of the information that is shown varies, depending on the type of module that is shown.

Set the source code mapping repository from the observations view by right-clicking in the Observation list view and selecting **Mapping Repository**, as shown in Figure 4-26.

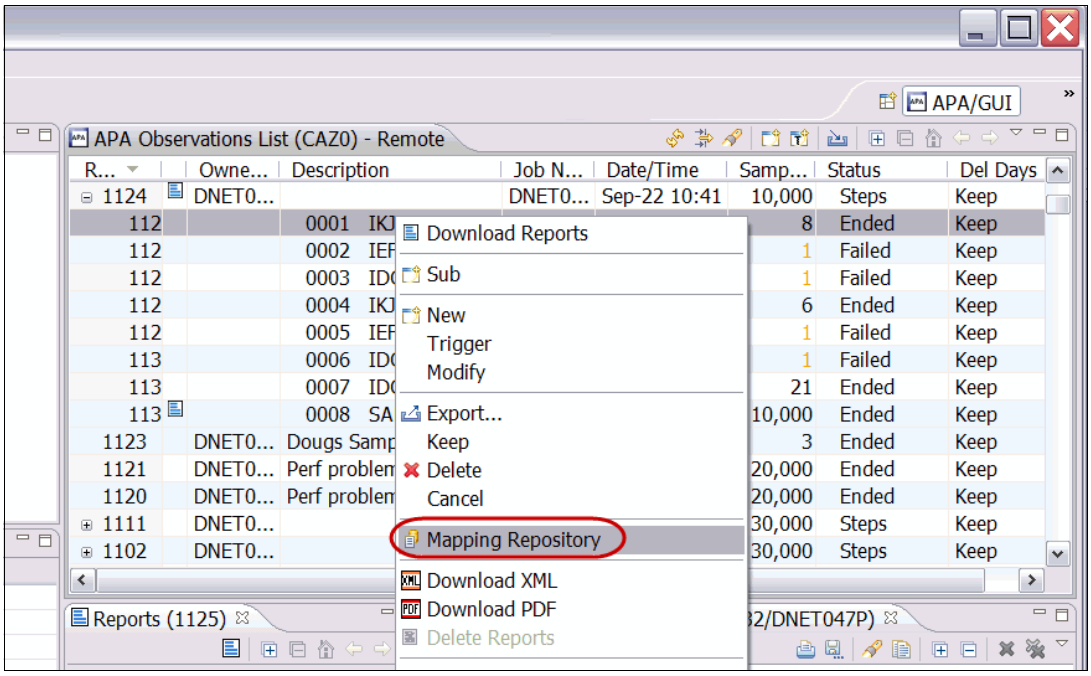


Figure 4-26 Mapping source code

The source code mapping repository list is used for all observations. The repositories contain the source information files, such as Sysdebug and Langx Files, as shown in Figure 4-27.

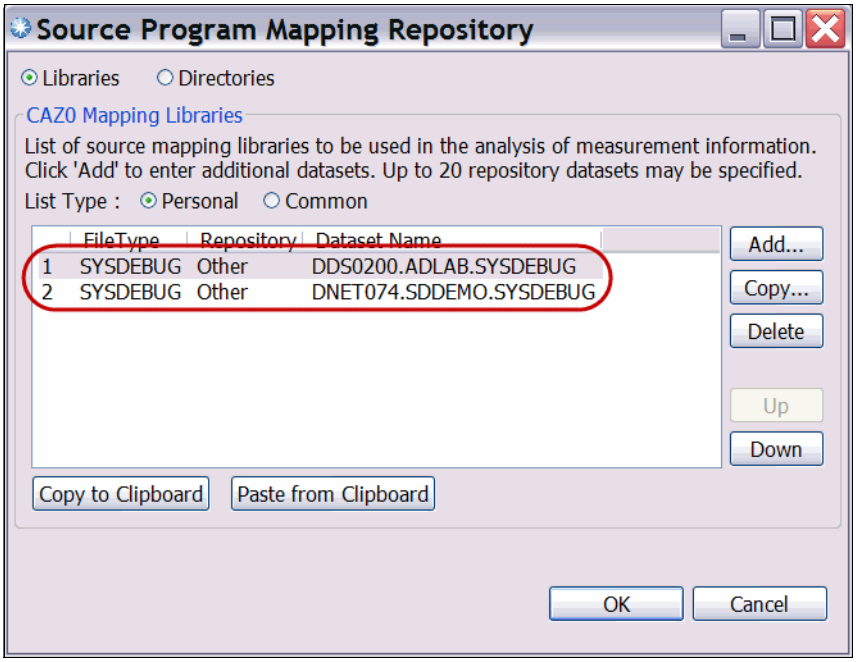


Figure 4-27 Source Program Mapping repositories

After a report is opened in the Reports view, you can obtain the source program information by selecting a specific line in the report that you want to view and right-clicking the link. Next, select **Source Program Mapping**. A separate window shows the source information and visual indication of the results from the APA observation, as shown in Figure 4-28.

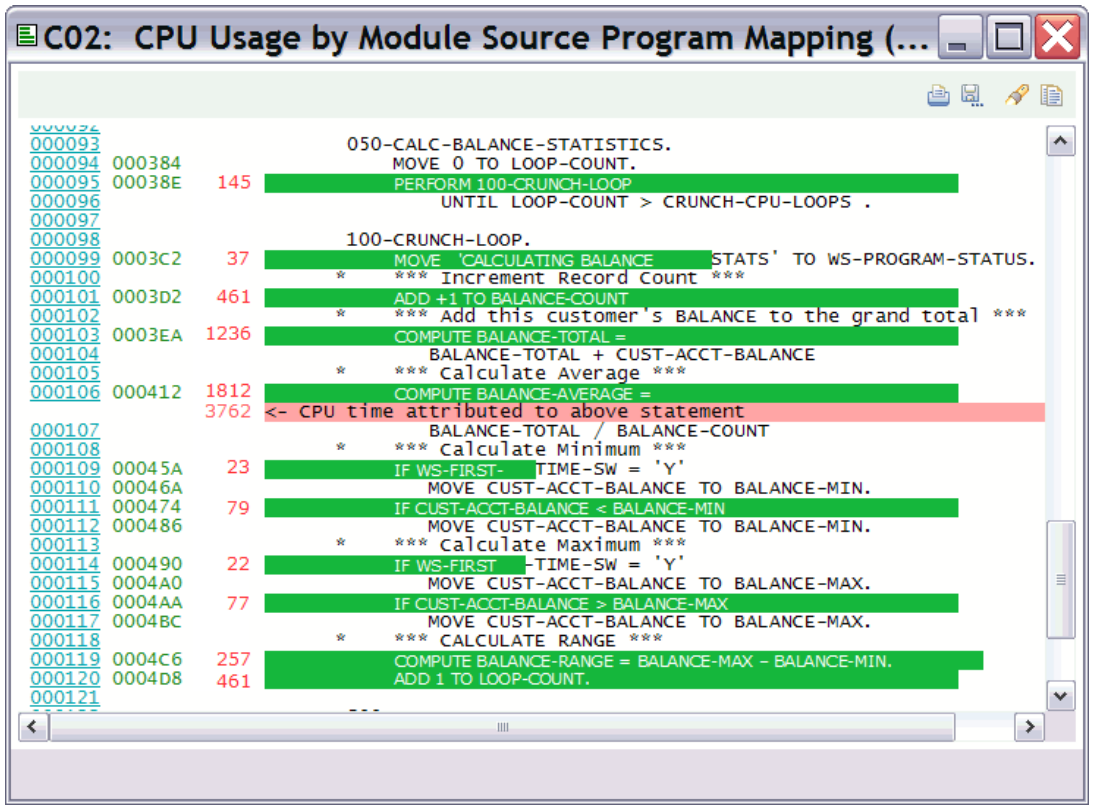


Figure 4-28 Source Program Mapping

4.3 File Manager

In this section, we describe topics that are related to the File Manager plug-in for Eclipse.

4.3.1 Introduction

By using the File Manager plug-in for Eclipse, you can view, manipulate, and browse data that is stored in various z/OS resources, including data sets, UNIX System Services, and IBM WebSphere MQ queues. You also can manipulate data that is stored in CICS resources and DB2 databases with File Manager V12[1].

The graphical user interface

The File Manager Eclipse plug-in provides a GUI alternative to the traditional z/OS ISPF interface. By using this GUI, developers of existing applications and system programmers can experience productivity gains and a quicker learning curve for new z/OS developers.

The File Manager plug-in, when integrated with File Manager for z/OS and the Eclipse-based platform, provides a set of tools for manipulating and browsing data that is stored in various z/OS resources. These resources include data sets, UNIX System Services files, and IBM WebSphere MQ queues. You also can manipulate data that is stored in CICS resources and DB2 databases with File Manager V12. The File Manager plug-in supports the following usage scenarios:

- Creation, modification, and updating of the following File Manager templates:
  - Create templates from COBOL, PL/I, and HLASM copybooks
  - Create dynamic templates if you do not have a copybook to describe your data
- Manipulation and browsing of data by using the File Manager editor. The editor provides the following data views:
  - Character display (text-based representation of data)
  - Formatted or table display by using a template
  - Single display by using a template (focused on the display of fields that are defined in a single record)
- Use of wizards to access utility functions to perform the following tasks:
  - Allocate, delete, and rename data sets
  - Copy, compare, and create data in resources, including z/OS data sets, WebSphere Message Queues, UNIX System Services files (USS) and CICS resources
  - View load modules
  - Copy, create data, export, import, and print data that is stored in DB2 databases

4.3.2 Showing the File Manager views

The File Manager Eclipse plug-in provides the recommended configuration to use the product. You can open the File Manager perspective by completing the following steps:

1. Select **Window** → **Show Perspective** → **Other** from the drop-down menu.
2. Select **File Manager** in the list of available perspectives. Click **OK**.

[1] Manipulation of DB2 resources is V12-only feature. Manipulation of CICS resources is also supported in V11. Apply PTF forAPAR PM47010.

4.3.3 Managing connections with the File Manager plug-in

Connections to the Problem Determination Tools common server are managed by using the z/OS Explorer's feature called Host Connections (for more information, see Chapter 2, "Introduction to the IBM Problem Determination Tools Plug-ins" on page 25). After you establish a connection in the Host Connections view, a corresponding connection is shown in the Systems Information view (see Figure 4-29). If you start the Add System action from the Systems Information view, a corresponding information connection also is registered in the Host Connections view.

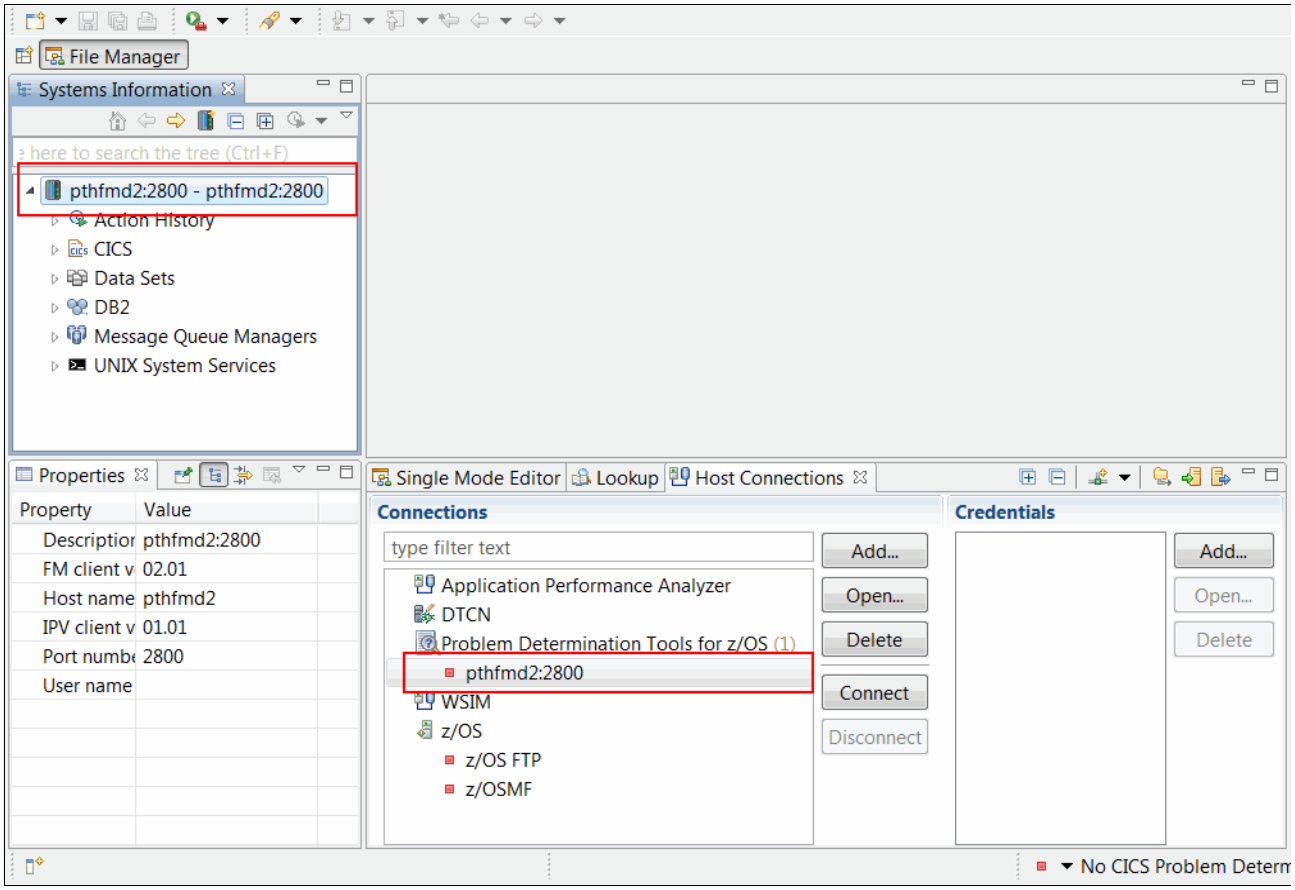


Figure 4-29 Systems information view is populated based on the information in the Host Connections view

Disconnecting from the File Manager Server

The File Manager plug-in operates by making one or more connections to each system in the Systems Information view on an as-needed basis. Each connection starts a new session under the user ID that is specified.

To manually close all active connections to a system, right-click the system, then click **Close Connections** from the pop-up context menu. All open connections are closed and the corresponding user sessions on the host end. Any query results are still visible. If you attempt to perform any operation that requires a connection (such as refreshing a query to get updated results), the plug-in automatically creates a connection.

To remove a system completely, right-click the system and click **Remove** from the context menu. All connections to the system close before the system is removed. When you remove a connection in the Systems Information view, a corresponding entry in the Host Connection

view also is deleted. Similarly, if you delete a PD Tools connection in the Host Connections view, a corresponding connection in the Systems Information view also is deleted.

4.3.4 Systems Information View navigation

Most of the functionality that is provided by the IBM File Manager plug-in for Eclipse is accessible from the Systems Information View. After a connection is created, you can call File Manager actions and add data set queries. Most actions in the Systems Information View are accessible through the context menu.

Systems Information context menu

The context menu is accessible by right-clicking an item in the view or by pressing Shift + F10. Depending on the selection in the view, a menu with a set of available actions is shown, as shown in Figure 4-30.

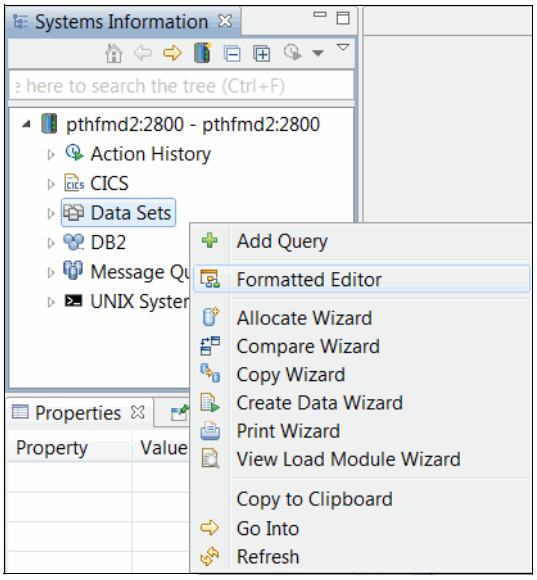


Figure 4-30 Sample context menu in the Systems Information view

A summary of the context menu actions is shown in Table 4-1.

Table 4-1 Context menu actions

Action	Description
Add Query	Adds a query to locate z/OS resources
Allocate Wizard	Starts a wizard to allocate various data sets or HFS file
Close Connections	Closes all connections to the selected system
Compare Wizard	Starts a compare wizard
Copy to clipboard	Copies the full names of currently selected items in the Systems Information view
Copy Wizard	Starts copy wizard to copy resources or data
Create Data Wizard	Starts wizard to populate resources with data
Create Template Wizard	Starts Template creation wizard to create a template
Delete	Deletes the selected resource on the z/OS system
Modify	Modifies existing queries and systems by using the relevant dialogs
Formatted Editor	Starts Formatted Editor Options dialog to start a view or edit session of the selected resource
Template Editor	Starts Template Editor for the currently selected resource
Print Wizard	Starts wizard to print contents of resources
Properties	Shows Eclipse's Properties view, which displays the properties of a selected item
Refresh	Refreshes the currently selected item in the view
Remove	Removes the selected item from the view
Update Login Details	Updates the user and password details for the selected system
Update Template Wizard	Starts the Template update wizard to update an existing template
View Error	Shows any errors that are associated with the currently selected item in the view
View Load Module Wizard	Starts the View Load Module wizard
Export	Export the contents of a DB2 table
Import	Import some data to a DB2 table

Adding Queries

Data Set Queries are used to narrow the list of resources that are shown, and to search for specific resources or groups of resources on the mainframe system. Queries can be specific (returning only a list of one or few results) or generic (returning many results). Complete the following steps to add a data set query:

1. Right-click in the Systems Information view and select **Add Query**.
2. By using the Add/Edit Data Set Query dialog, you can specify queries to match data sets based on the following data set name patterns:
   - % matches a single character
   - * matches multiple characters in a single qualifier
   - ** matches any characters in any number of qualifiers

   You also can filter data sets by their type. When a VSAM data set type is selected, its associated INDEX or DATA component data sets also are shown. In this example, all of the data sets that include a high-level qualifier of TSS09 are included. Click **OK** to add the data set query, as shown in Figure 4-31.

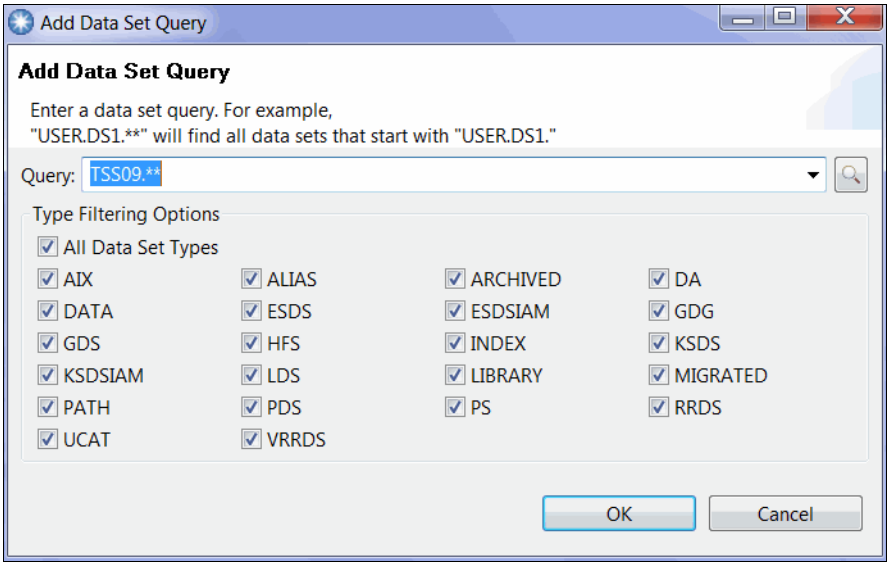


Figure 4-31 Add data Set Query

3. The query is added to the Systems Information view. Expand the query folder to see the results, as shown in Figure 4-32.

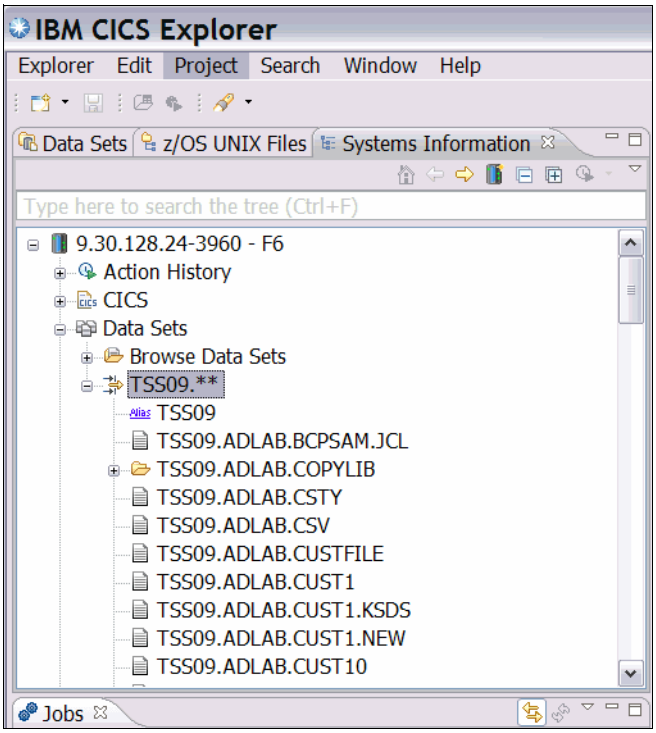


Figure 4-32 Expanded Query results

By using the Add/Edit Message Queue Query dialog, you can create a query to locate WebSphere MQ queues for the specified WebSphere MQ manager at the specified system. A pattern can be specified for the queue name, but not for the WebSphere MQ manager. The supported wildcards are “%” (single character) and “ * ” (multiple characters).

By using the Add/Edit USS File Query dialog, you can create a query to locate UNIX System Services Hierarchical file system files on a specified system. The query can show all of the files in a particular directory, or a subset of the files in the directory by specifying a type filter or by specifying wildcards. (Although all file types are collectively referred to as *HFS Files* or *USS Files*, the type *File* represents a standard editable file.)

Important: The queries `/u/fmn*/` and `/%/fmn/` are not valid queries because File Manager does not find files from multiple directories. Paths that include % or * characters are not supported.

The Add/Edit CICS Resource Query dialog can be used to create queries to locate CICS File, Transient Data Queue, or Temporary Storage Queue resources (collectively known as *CICS Resources*) on a particular CICS System on a particular system. A pattern can be specified for the resource name, but not for the CICS System (which is referred to by the CICS Application ID).

Action history

By using of the action history, you can repeat, modify, and duplicate previously called File Manager wizards. Complete invocation of any File Manager Utility Function Wizards automatically is stored as the top-most entry in their respective group under Recent Actions. An action can be started through the context menu or by double-clicking the action.

Starting an action silently (that is, without showing any dialogs) starts the File Manager Utility Function with the Action Configuration unless the last execution resulted in an error. The exception to this rule is the Delete action, which always shows a confirmation dialog. While in an error state, starting an action causes the appropriate wizard or dialog to open, as shown in Figure 4-33.

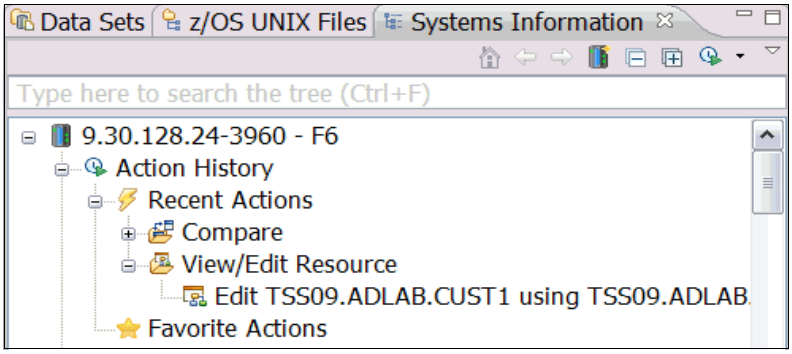


Figure 4-33 Systems Information view Action History

Additionally, the 10 most-recently run actions can be started directly from the toolbar drop-down menu, as shown in Figure 4-34.

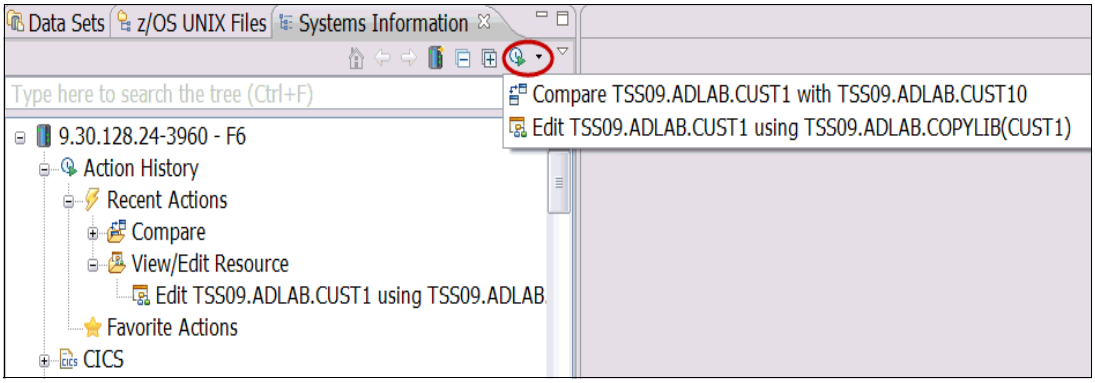


Figure 4-34 Action history menu

4.3.5 Opening a data set

You can browse or edit various z/OS resources by using the File Manager editor. The editor is started from the Systems View by either right-clicking the resource and selecting **Formatted Editor** from the context menu of a supported resource, or by double-clicking a supported resource in the Systems Information view, as shown in Figure 4-35.

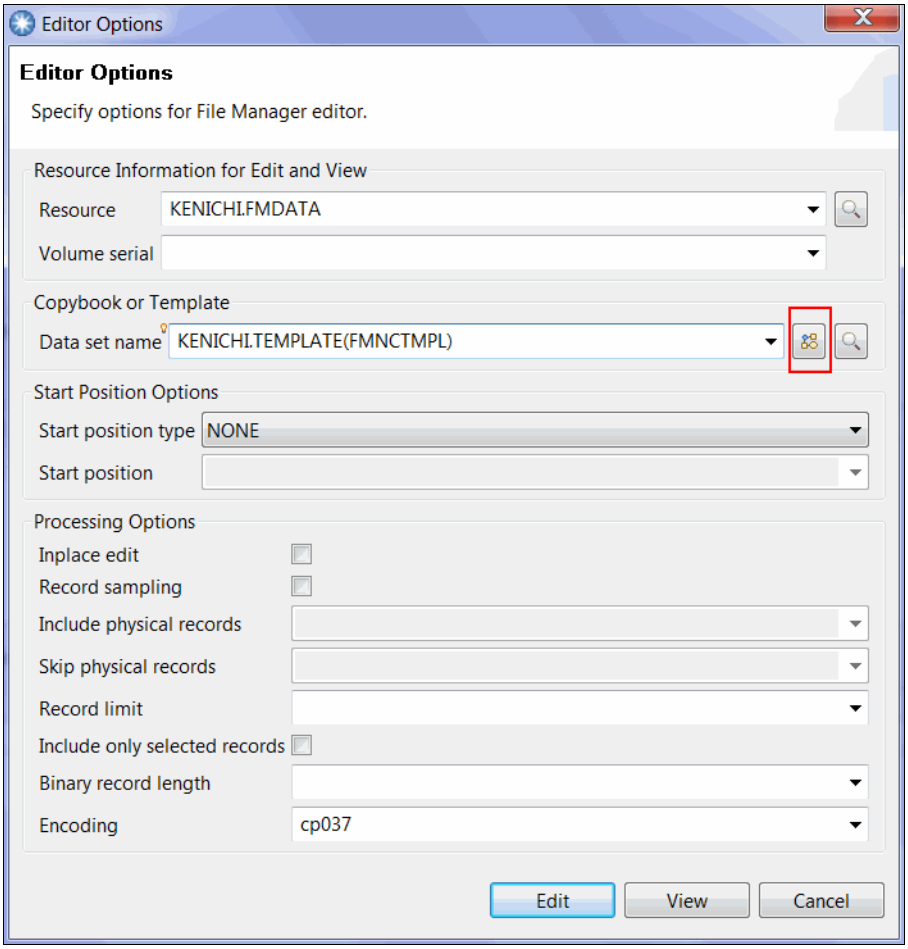


Figure 4-35 Editor Options window

When the editor is started by using one of these methods, the editor's option window is shown. In this window, you can specify various options that control how the editor is started. The following main options are available:

- Resource: The z/OS resource that is used to browse or edit the z/OS resource.
- Copybook or Template data set name: The name of a copybook or template to use to format the data that is stored in the selected resource.
- Encoding: The encoding that is used to convert characters for text fields (AN) in the selected resource.

For a full explanation of the available options, see this website:

http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp?topic=/com.ibm.iea.fileman/fileman/6.1z/TrainingEducation/Base/FMBaseIntro/FMBv10s01Introduction/player.html

Based on the options that are specified, the File Manager editor is started against the selected resource. The initial look of the File Manager editor depends on whether a copybook or template was specified. If a copybook or template was specified, the editor is started in Formatted Mode; otherwise, the editor is started in Character Mode.

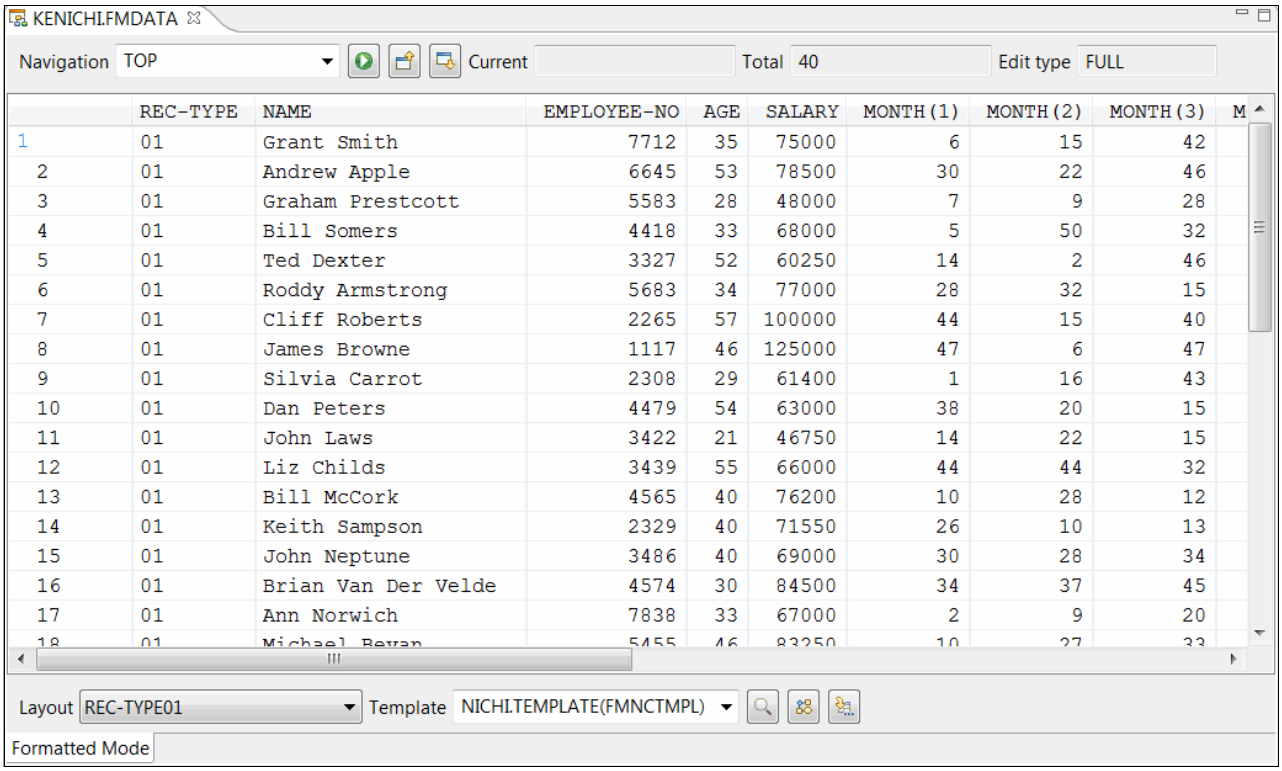
Lookup button: Use the Lookup button for quick retrieval of resources from the host. To use this feature, enter a partial resource name in the resource field and click **Lookup**. The lookup window is shown, the partial resource name is entered in the filter field at the top of the window, and the matching results are loaded. Use *, **, and % as wildcards in the filter.

Template editor invocation: There also is a button to look up a template to use and a button to start the Template Editor (see Figure 4-35 on page 99). After a template to use is specified, you start the template editor before the formatted editor is started. By using this feature, you ensure that the selected template is the template you want to use. You also can specify additional criteria for the edit session. The Template Editor button is provided throughout the File Manager product for convenience.

4.3.6 Editor Display mode

The File Manager editor provides the following display modes that are used for data that is stored in various resources:

- Formatted/Table mode: Displays data in a table, and each column represents a field within a layout of a template or copybook, as shown in Figure 4-36.



	REC-TYPE	NAME	EMPLOYEE-NO	AGE	SALARY	MONTH(1)	MONTH(2)	MONTH(3)	M
1	01	Grant Smith	7712	35	75000	6	15	42	
2	01	Andrew Apple	6645	53	78500	30	22	46	
3	01	Graham Prestcott	5583	28	48000	7	9	28	
4	01	Bill Somers	4418	33	68000	5	50	32	
5	01	Ted Dexter	3327	52	60250	14	2	46	
6	01	Roddy Armstrong	5683	34	77000	28	32	15	
7	01	Cliff Roberts	2265	57	100000	44	15	40	
8	01	James Browne	1117	46	125000	47	6	47	
9	01	Silvia Carrot	2308	29	61400	1	16	43	
10	01	Dan Peters	4479	54	63000	38	20	15	
11	01	John Laws	3422	21	46750	14	22	15	
12	01	Liz Childs	3439	55	66000	44	44	32	
13	01	Bill McCork	4565	40	76200	10	28	12	
14	01	Keith Sampson	2329	40	71550	26	10	13	
15	01	John Neptune	3486	40	69000	30	28	34	
16	01	Brian Van Der Velde	4574	30	84500	34	37	45	
17	01	Ann Norwich	7838	33	67000	2	9	20	
18	01	Michael Bevan	5455	46	83250	10	27	33	

Figure 4-36 Formatted editor mode

- Character mode: Displays data as text, as shown in Figure 4-37.

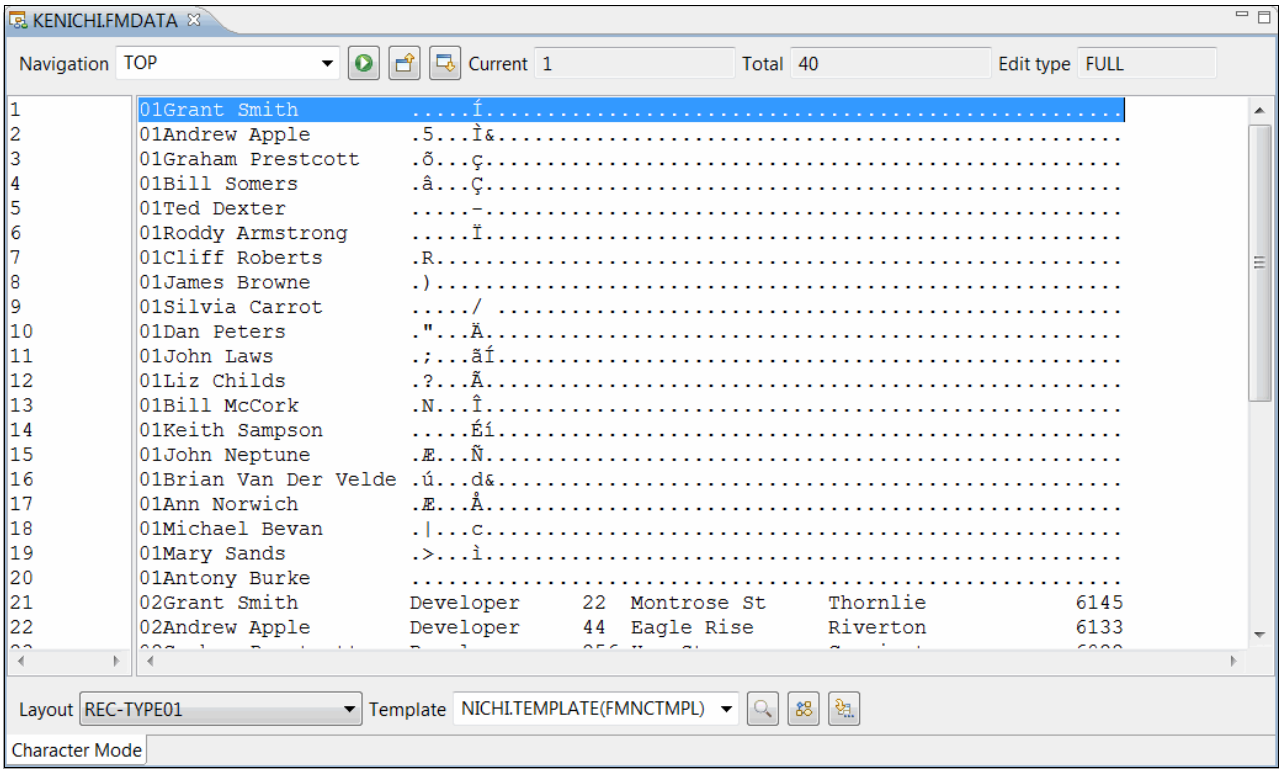


Figure 4-37 Character Editor Mode

- Single mode: In a separate view, data about the currently selected record in the editor is displayed in Formatted or Character mode, as shown in Figure 4-38.

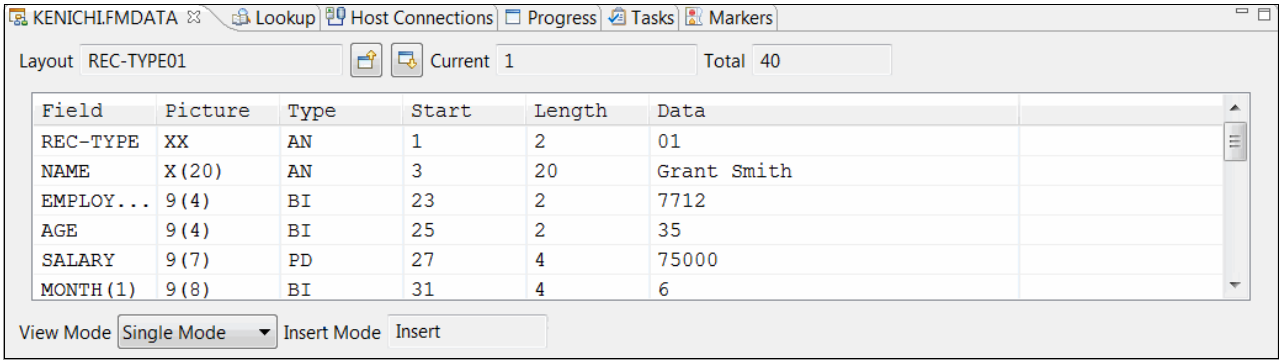


Figure 4-38 Single Mode Editor

The separate Single Mode view can display the selected record in Dump Mode, the dump format of a record in hexadecimal, and its EBCDIC translation, as shown in Figure 4-39.

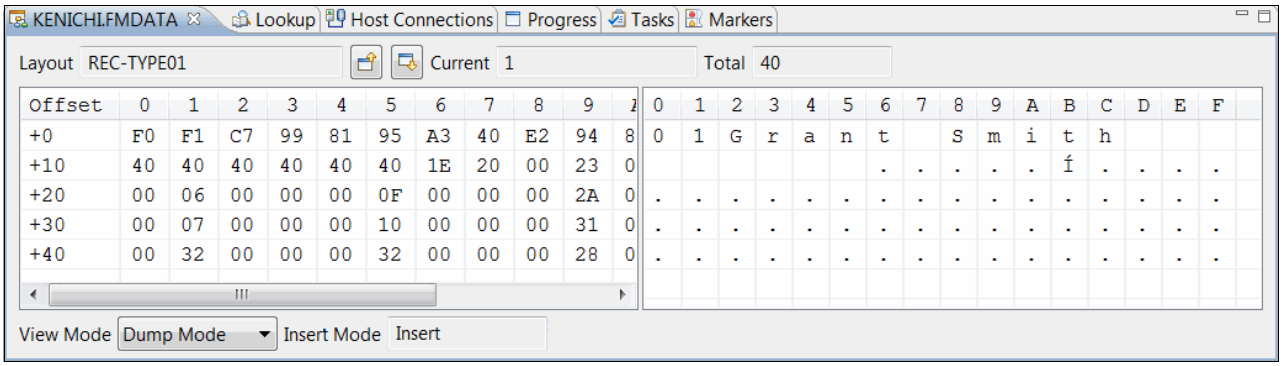


Figure 4-39 Single Mode Dump display

4.3.7 Working with the File Manager Editor

The File Manager Editor provides navigation, basic edit functions, and additional edit capabilities through toolbars and context menu items.

Navigation

The File Manager editor caches only a subset of records locally for display. The number of cached records is controlled in the File Manager plug-in preference page. This configuration allows efficient transfer of data from mainframe to your desktop machine, particularly when you are working with a resource that contains a large number of records.

You can maneuver and control your position within File Manager by using the Navigation toolbar. This toolbar contains the currently selected record, and the total number of records in the data set. The Page Up, and Page Down buttons move the currently displayed records up or down to a new set of cached records, which is set in the File Manager plug-in preferences page.

There also is a navigation field in which you can enter the following commands, as shown in Figure 4-40:

- Top: Moves the displayed records window to start with the first record in the data set.
- Bot or Bottom: Moves the displayed records window to end with the last record in the data set.
- Up nn or -nn: Moves the displayed records to begin with a record that is nn above where it is displayed.
- Down nn or +nn: Moves the displayed records to end with a record that is nn below where it is displayed.
- Locate nn: Positions the displayed records to begin with the nn record.

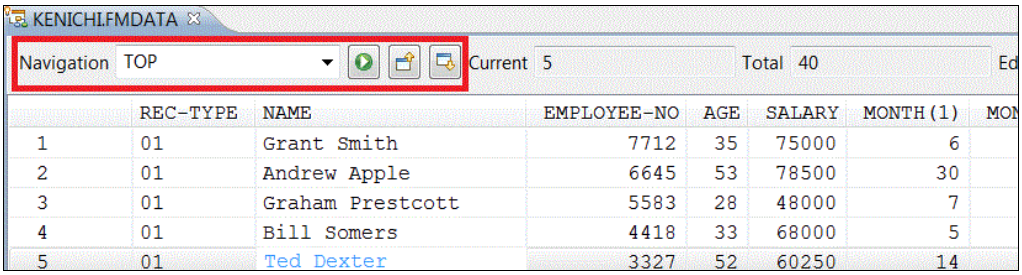


Figure 4-40 Editor navigation bar

Additionally, you can locate columns within the current edit session. This feature is available only when a template is used to format the data and the records are displayed in formatted mode. Right-click in the File Manager Editor and select **Locate Column**. The Locate Column window is shown, and you can select the column from the list of fields that are available in the currently selected layout of the template that is used for the current edit session. After a field name is selected from the list, the editor tries to adjust the position of the table so that the selected field or column is visible in the editor, as shown in Figure 4-41.

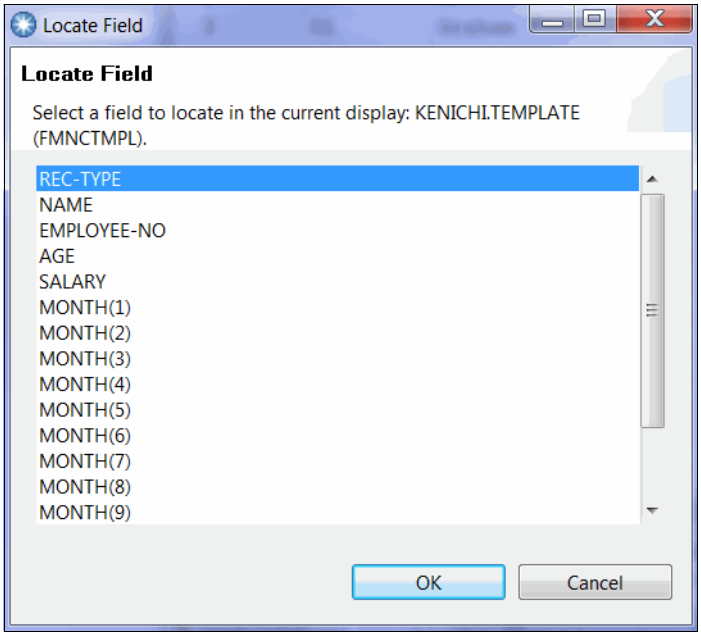


Figure 4-41 Locate Field window

Find/Replace

By using the Find/Replace feature, you can search for a specific pattern (by a string literal or hex value) within the currently edited resource. You also can issue a replace command that replaces the occurrences of searched patterns with the specified replacement pattern. Right-click in the File Manager Editor and select **Find/Replace**, the Find/Replace window is opened. Figure 4-42 on page 104 shows the options that can be specified by using Find/Replace dialog.

Figure 4-42 Find/Replace window

The following parameters are specified in the Find/Replace window:

- Find: String literal or pattern to search, or a hex or picture value.
- Replace with: String literal or pattern to search, or a hex or picture value to replace search strings. This parameter is only applicable for an edit session.
- Field or column Selection (Search range): Sets the search scope. The search string is searched for ONLY within the specified fields or column range.
- Navigation: Controls how the result of search is reported.
- Others: Find/replace option that you use to highlight strings that do not match the specified pattern. It is best to specify a search range parameter to narrow the scope of the search.
- Match Options: Controls the position of the search string
- Search by Hex Value: Specifies that the search string is entered as its hexadecimal value.
- Search Scope Options: Specifies the scope of search. In particular, it controls if the search must occur within or outside of excluded records. See Figure 4-6 on page 76 for more information.

For a full description of these options, see the *IBM File Manager for z/OS, User's Guide and Reference*, SC19-2857-00.

After a Find/Replace command is issued, the result is reflected in the editor contents. Figure 4-43 shows the File Manager editor after successfully issuing a Find command to the current edit session (the fields that contain the search pattern are highlighted). The Find/Replace dialog remains active when the current edit session is open, so you can interactively specify search commands.

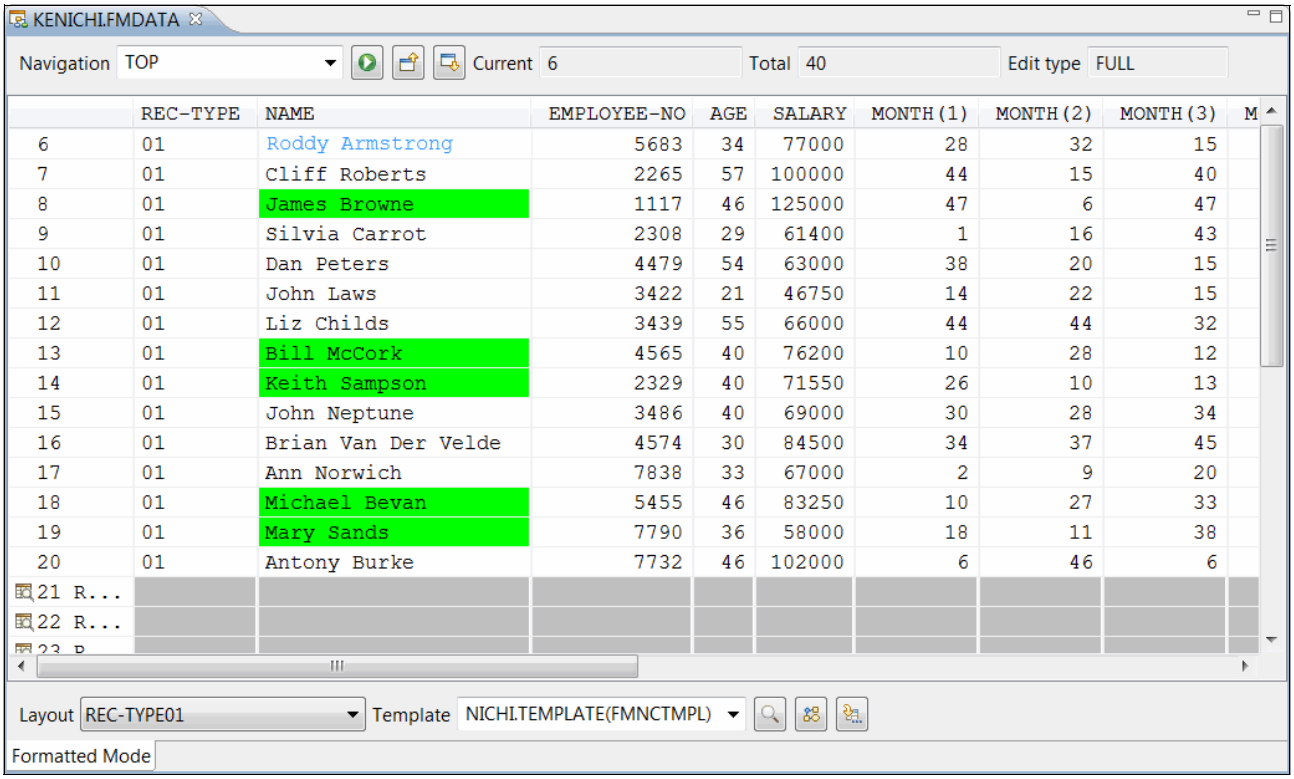


Figure 4-43 Result of Find command execution

After the Replace command is successfully run, the records are modified in the editor for display. However, the changes to records are not committed to the edited resource until a Save is run explicitly.

Additional editor actions

Along with the basic edit capabilities, the File Manager editor allows many of the same functions that are found in the green screen interface of File Manager. The following functions are included in the editor context menu:

- Copy records

  This function copies the currently selected records in the editor. The copied records are available for Paste. Grouped records (such as excluded records), grouped cleared records, and grouped suppressed records are not available.

- Cut records

  This function cuts the currently selected records in the editor. This function is the equivalent of performing Copy action followed by Delete action against the currently selected records. Grouped records are not available for this function. After records are cut, they are removed from the editor for display. However, the records are not removed from the original source unless a Save is performed explicitly.

- Paste records

  This function pastes the copied or cut records into the currently edited resource. This function is only available in an edit session when the resource that is edited allows insertion of new records. In the Paste window, if you specify Paste records at the top option in the Paste Records window, the records are inserted at the beginning of the currently edited resource. Otherwise, the records are inserted after the currently selected records in the editor. The records are not saved to the edited resource until a Save is explicitly run.

  In the Paste Records window (as shown in Figure 4-44), you can specify the number of times the selected records must be pasted into the current edit session (that is, repeated). The maximum number of repeats is 1000.

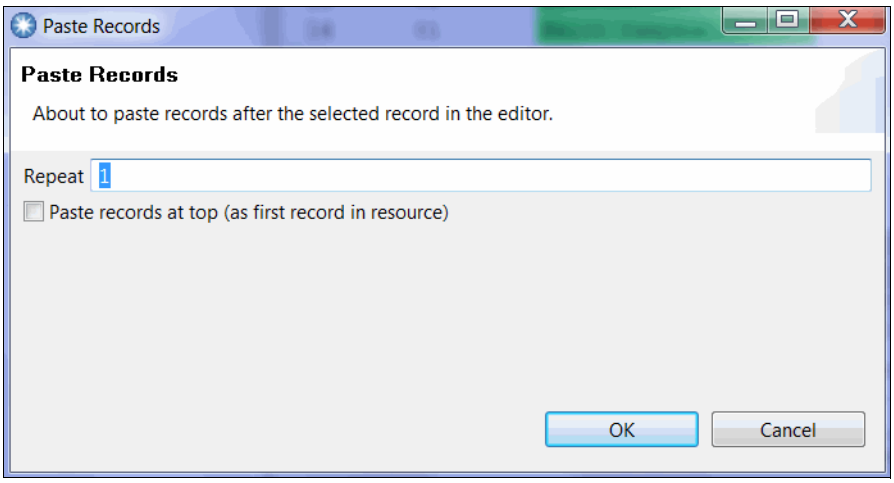


Figure 4-44 Paste Records window

- Delete records

  This function deletes the currently selected records in the editor. As shown in Figure 4-45, the Delete all records option deletes the entire contents of the resource. Grouped records are not available for this function. After records are deleted, they are removed from the editor for display, but not deleted from the original source until a Save is performed explicitly. The delete confirmation window provides the option to delete all records.

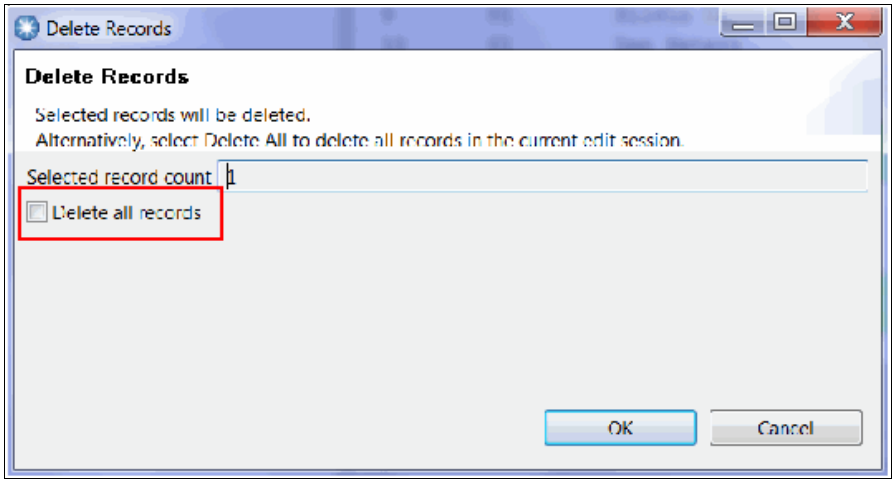


Figure 4-45 Delete Records window

- Exclude records

  This function excludes some records temporarily (for the current edit session only) from display. The criteria for excluding records are specified by using the Exclude Options window as shown in Figure 4-46. The Exclude String Pattern is string pattern that represents a criteria for Exclude action. You can specify a pattern by using a string literal or a hex value. The field or column Selection sets the Search Range, and you can narrow the search scope and only search for the specified pattern within the specified field or column range. Under Search Options is the option to Exclude all records, which removes all of the records from view. This feature can be useful when combined with the Find command. Exclude Options controls how to exclude pattern matching is performed. Match Options controls how a pattern is matched against the data in a record.

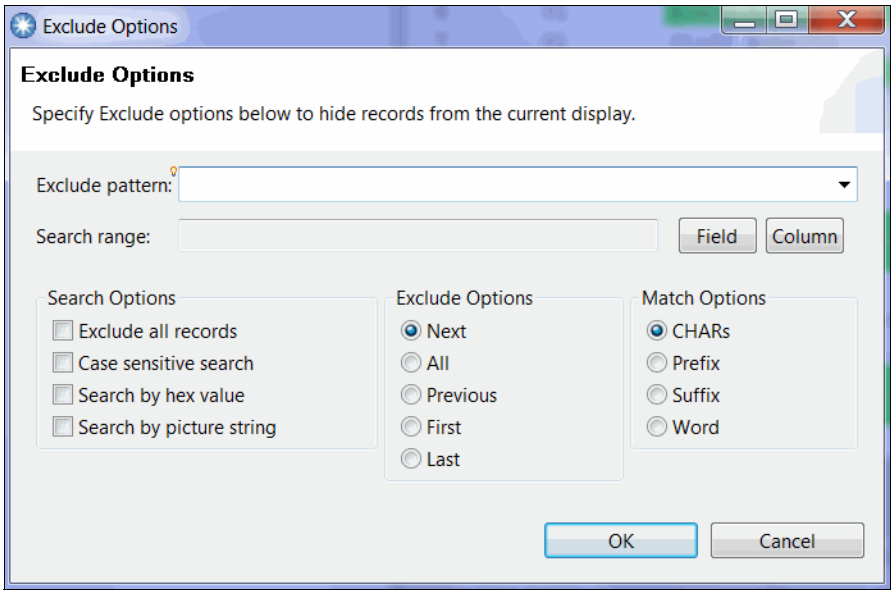


Figure 4-46 Exclude Options window

- Reset exclude options

  This function resets the current Exclude options. All previously excluded records are displayed again.

- Insert new records

  As shown in Figure 4-47, this function inserts a specified number of new records into the current edit session. This function is only available in edit sessions and data set types that allow new records to be inserted. You can specify the number of new records to insert, and if you select Insert records at the top option in the window, the specified number of new records is inserted at the top of the editor. Otherwise, the new records are inserted after the selected record or after the first record from a group of selected records. The inserted records are inserted to the editor for display, but not saved until a Save is run explicitly ().

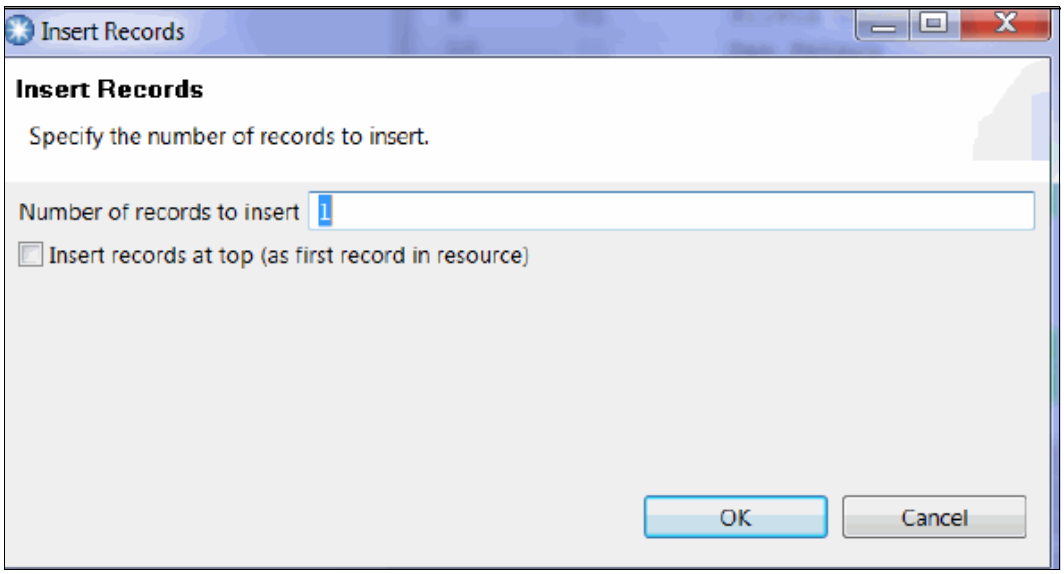


Figure 4-47 Insert records window

- Hex On/Off

  This function toggles hex lines on or off.

- Save Records

  This function saves the modified records; the changes are made to the resource on a permanent basis. This function is applicable only in an edit session and becomes available only when the editor contents are modified.

- SaveAs Records

  This function allows a user to save the contents of a currently selected resource to a different resource on the host. When this function is used, the user is presented with a window that allows them to specify the target resource. Only records that are visible from the show options are saved.

- Sort Records

  This function sorts records that are based on the specified criteria. The sorting criteria is specified by using the Sort record window, as shown in Figure 4-48. For Sort by column number and Sort by field reference, up to five criteria can be specified. Depending on the edit session, the following sorting methods are available:

  – Sort by column number: Sort by column number in ascending or descending order.
  – Sort by field reference: Sort by values in a selected field in ascending or descending order. This option is available only when a template or copybook is specified for the current edit session.
  – Sort by KSDS key: Sort by value of the KSDS key in ascending or descending order. This option is available only when you are editing a KSDS file.

  Click **Add Criteria** to add the criteria to the sort action. By selecting Sort by column number, you can indicate the columns that are to be sorted. By selecting Sort by field reference, you can select the fields to be sorted. By selecting Sort by KSDS key, you can sort by the KSDS key value ().

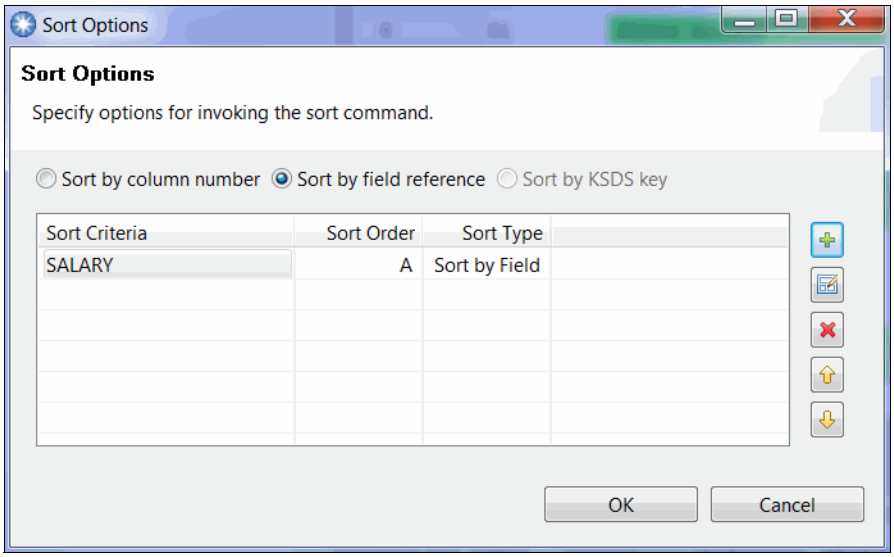


Figure 4-48 Sort options window

- Show options

  This function controls how records are grouped (grouping options) and which records must be displayed to the user (shadow line options). The following options are available:

  – Group NOT SELECTED records: Records are considered NOT SELECTED when they do not belong to a specific layout of a template that is used for the current edit session. When this option is specified, these records are grouped. Otherwise, these records are displayed as NOT SELECTED records in the editor.
  – Group SUPPRESSED records: Records are considered SUPPRESSED when they do not belong to the current layout of a template that is used for the current edit session. When this option is specified, these records are grouped. Otherwise, these records are displayed as separate SUPPRESSED records in the editor.

- Shadow line options

  The following options control whether a shadow line must be displayed to represent a specific type of grouped record:

  - Display shadow lines for excluded record groups: Excluded records are always grouped. If this option is selected, a shadow line is displayed in the editor so that the user can clearly see how many records are excluded.
  - Display shadow lines for cleared record groups: When Group NOT SELECTED records option is specified, NOT SELECTED records are grouped. When this option is specified, a shadow line is displayed in the editor, which makes it easy to see how many records are NOT SELECTED.
  - Display shadow lines for suppressed record groups: When a Group SUPPRESSED records option is specified, SUPPRESSED records are grouped. When this option is specified, a shadow line is displayed in the editor so that the user can clearly see how many records are SUPPRESSED.

  These options are specified by using the Show and Shadow Lines Options window, as shown in Figure 4-49.

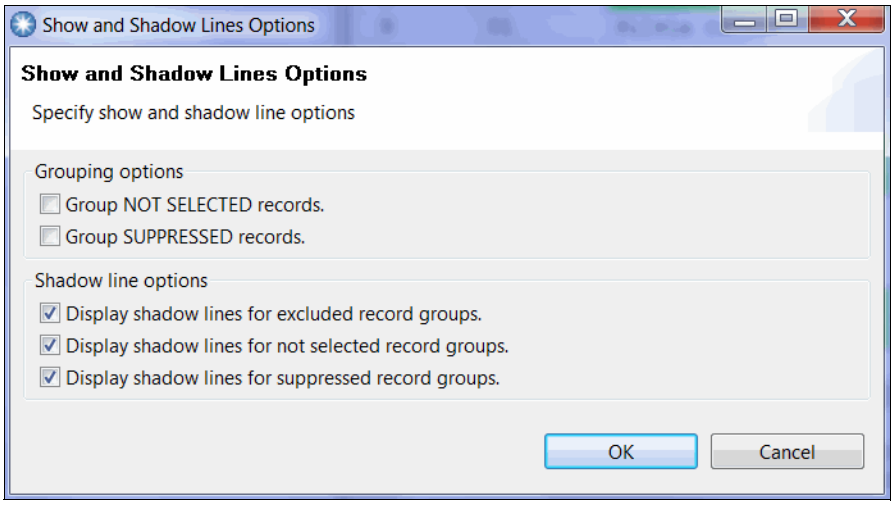


Figure 4-49 Show and Shadow Lines options window

- Switch editor mode

  This function switches between Formatted (table) mode if a template or copybook is applied and Character mode.

- Validate Records

  This function only validates changes to the currently edited records. If any errors exist, they are reported to the user. If validation is successful, or if the user chooses to ignore the errors, the changes are reflected in the edit session. However, changes are not saved to the currently edited resource until a Save is explicitly run.

Editor limitations

The File Manager editor provides many of the same functions as the green screen interface. However, the following limitations are inherent to the Eclipse-based platform:

- Column limitations

  Each column in formatted and single display displays only up to 259 characters. This limitation is imposed by the native Windows implementation of tables, which the Eclipse platform uses. The limitation also is recognized as a known issue on the Eclipse platform.

- Workaround

  Split a column that needs more than 259 characters into multiple columns by using a File Manager template. The data also can be displayed in Character mode.

- Column header truncation

  When the total number of characters exceeds approximately 3,000, column headers start to get truncated.

- Workaround

  Use a File Manager template to reorder or hold columns so that columns of your interest are shown first. Also, use the Locate Column action to locate a column. The column header text is missing, but the values are still visible.

- Alignment of text value and corresponding hex value display

  Depending on the type of records of resource that are being edited, the alignment of the text value display and the corresponding hex value display can be mis-aligned. This configuration is possible because some text values occupy multiple bytes for the corresponding text in UNICODE (such as, DBCS characters, UNICODE characters).

- Undo/Redo limitations

  The Undo and Redo functions are available only while you are editing in Character Mode. Additionally, enabling or disabling hex view while you are editing in Character Mode clears the undo buffer.

Working with DB2 resources

You can edit and view DB2 resources by using the File Manager formatted editor. The functionality of the formatted editor is almost identical to the functionality of the formatted editor while you are working with FM/Base resources, such as data sets, UNIX System Services files, CICS files, and WebSphere MQ queues.

The major difference is the editor options window, as shown in Figure 4-50, when you select and run the formatted editor action against a DB2 resource in the Systems Information view.

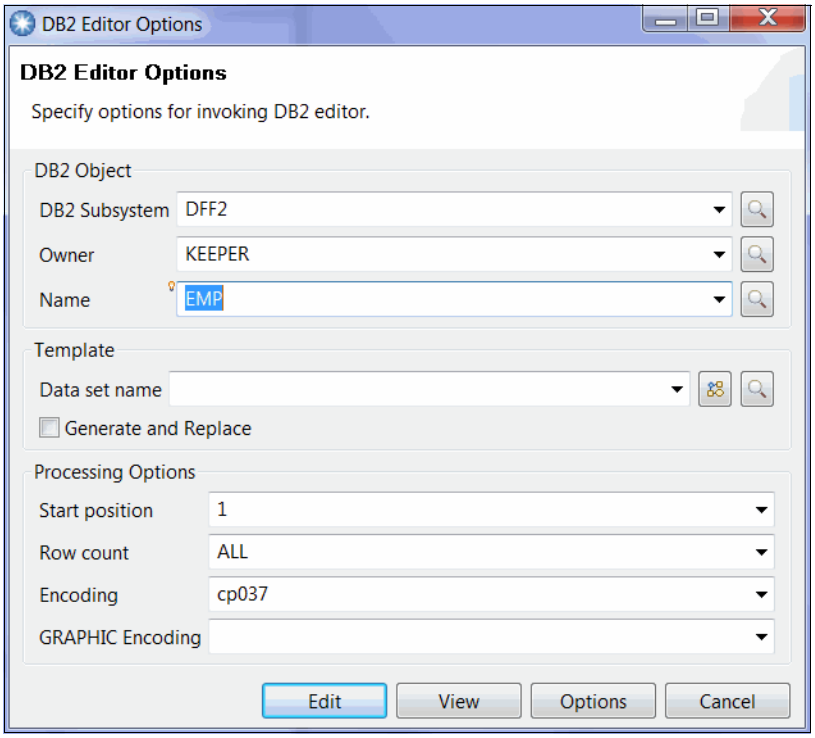


Figure 4-50 DB2 editor options window

Figure 4-51 shows the formatted editor when a DB2 resource is manipulated.

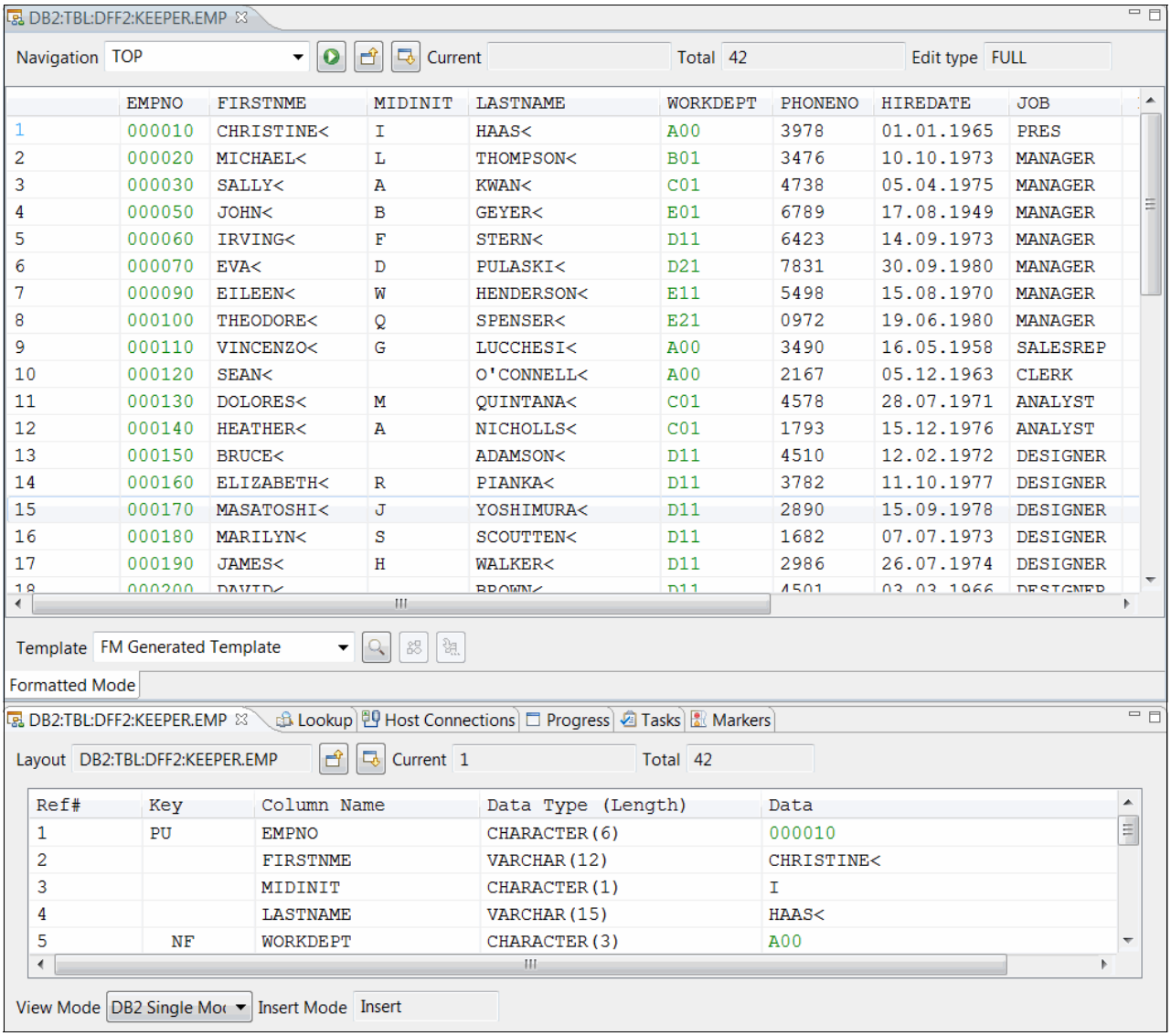


Figure 4-51 DB2 formatted display and single mode display by using FM's formatted editor

4.3.8 Working with templates

You can create a template by using copybooks or create a dynamic template by specifying layout of a data source interactively.

Creating template by using copybooks

You can create a template from COBOL, PL/I, or HLASM copybooks. Complete the following steps to create a template:

1. Create a query in the Systems Information view and look up a copybook from which you want to create a template.
2. Select a copybook member and start the template editor, as shown in Figure 4-52.

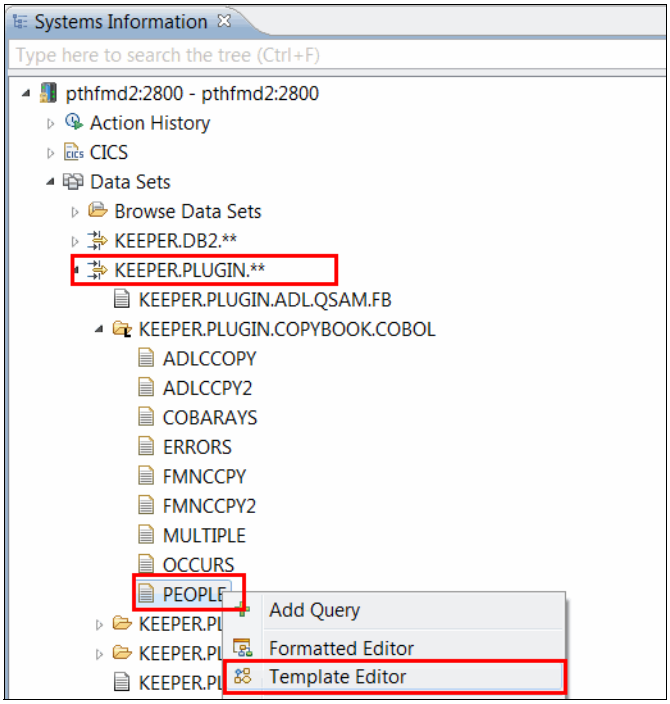


Figure 4-52 Calling Template Editor against a COBOL copybook to create a template

3. If the selected copybook compiles successfully, the template editor displays the contents of the new template, as shown in Figure 4-53. By default, the template editor is displayed in the dialog mode[2]. You can switch to the editor mode by clicking **Editor**.

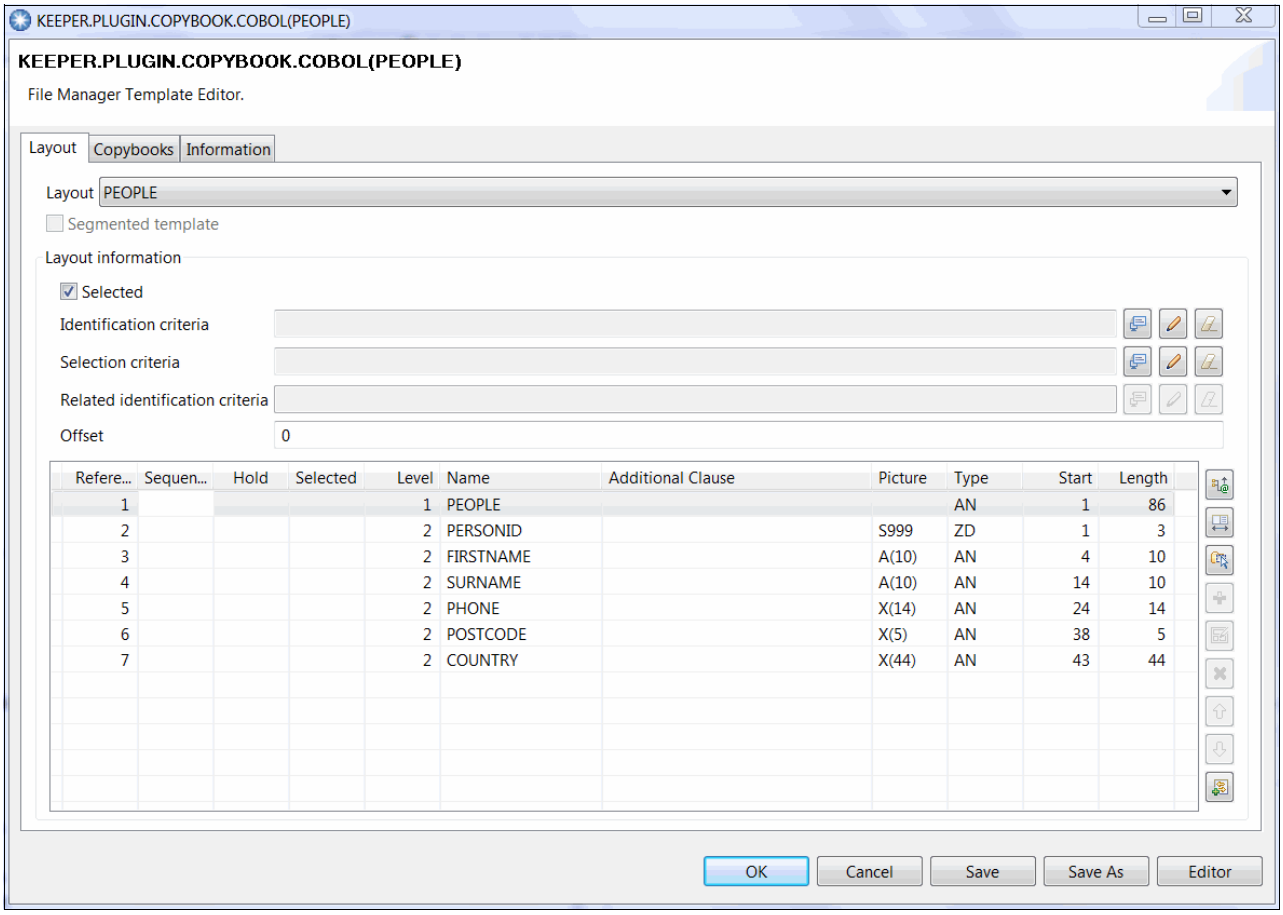


Figure 4-53 Template editor Layout tab that shows the fields and criteria of the current layout

4. If the selected copybook contains one or more errors, a window indicates the failure. In such a situation, compiler listing is useful to help you diagnose the problem. Click **Compiler Listing** to retrieve the compiler listing, as shown in Figure 4-54.

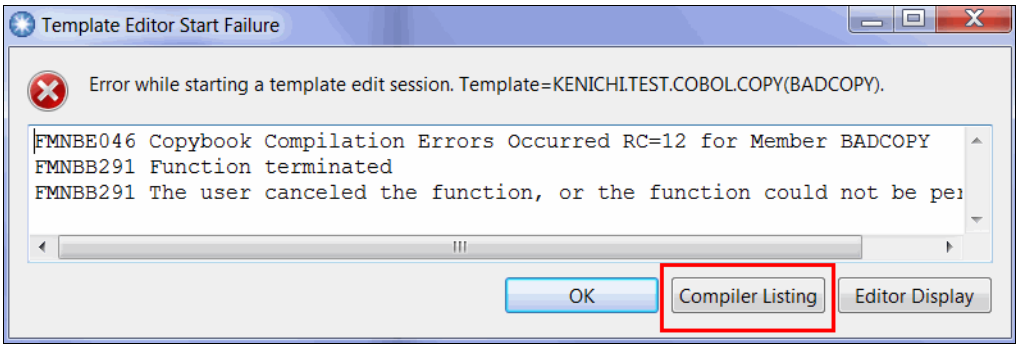


Figure 4-54 Base template creation error

[2] The dialog mode allows the invocation of the template editor throughout the FM product. For example, you can view and manipulate a template from a copy wizard without exiting the wizard.

5. Depending on the nature of the compiler error, you might have to adjust the compiler settings. This adjustment is made in Preference page. To access the page, click **Window → Preferences.** Select **File Manager → Compiler Options**, as shown in Figure 4-55. The compiler options are maintained per connection, so you must select the correct connection to set the compiler options.

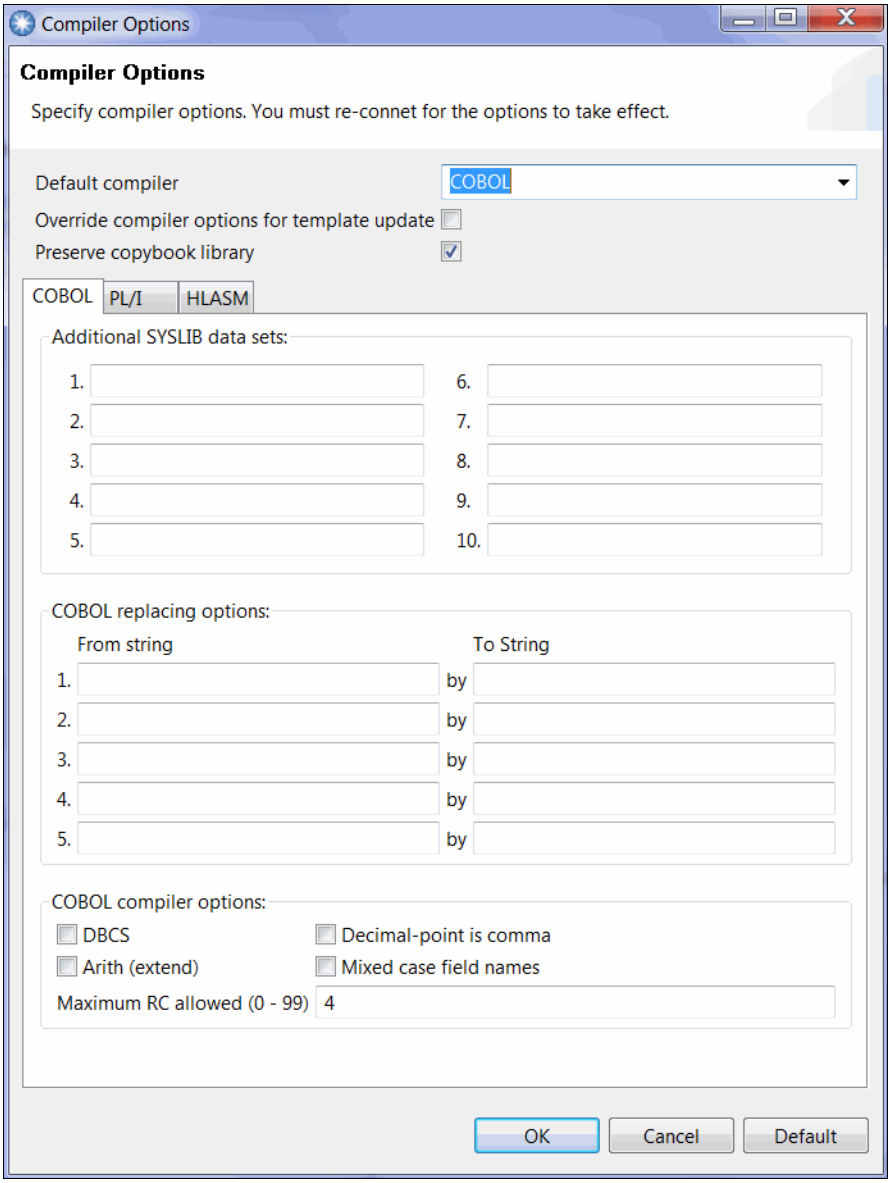


Figure 4-55 File Manager Compiler options dialog

6. You can add more layouts to the current template by adding more copybooks, as shown in Figure 4-56.

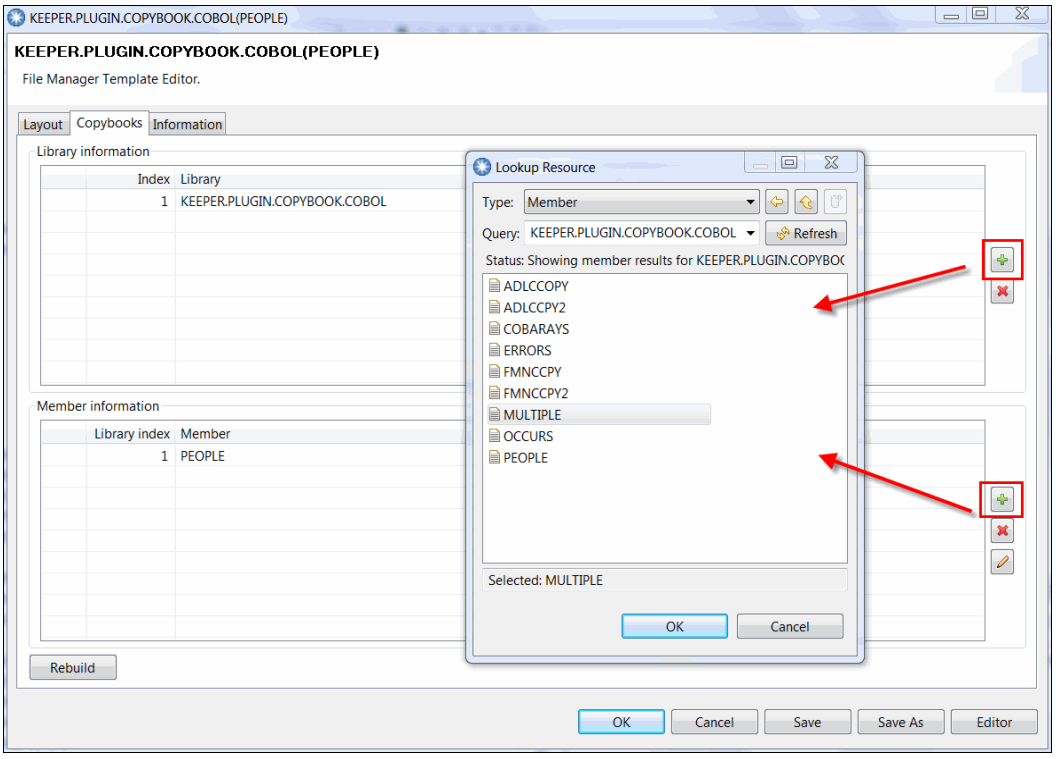


Figure 4-56 Adding more copybooks to a template

7. After you are finished, click **Save As** to save the template.

Creating dynamic template

Complete the following steps to create a dynamic template:

1. Select a data set in the Systems Information view and select **Create Dynamic Template** from the context menu, as shown in Figure 4-57.

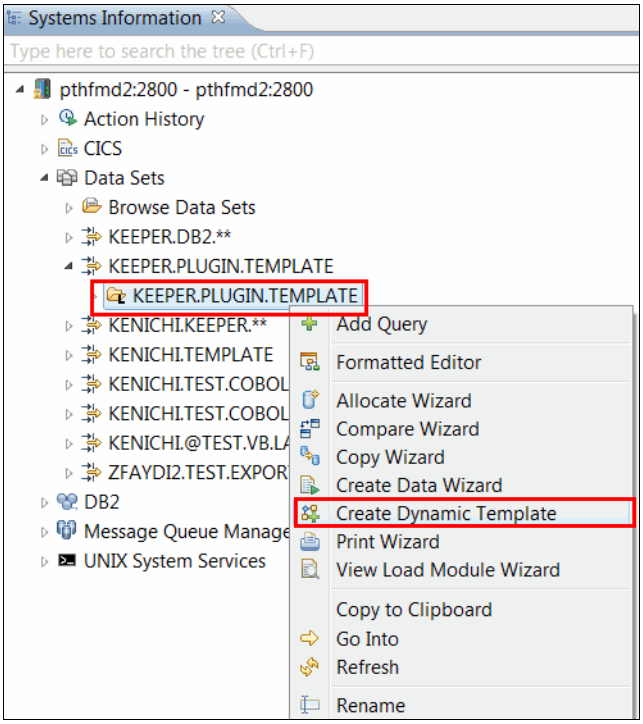


Figure 4-57 Starting the Create Dynamic Template wizard

2. In the Create Dynamic Template window, specify the name for the new dynamic template, as shown in Figure 4-58. You can specify a model template name where you can select a template, import its layout, and create a dynamic template that is based on the selected layout.

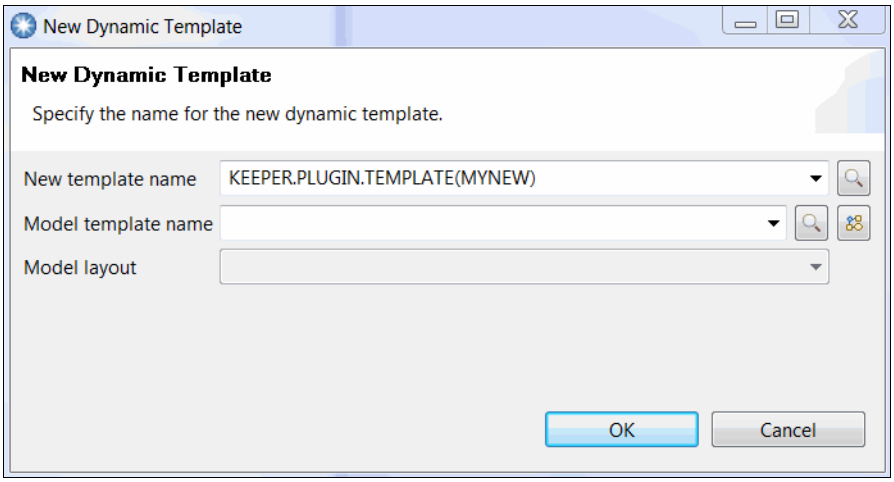


Figure 4-58 Dynamic template create dialog

3. Click **OK** to finish. After the dynamic template is created, its contents are displayed in the template editor. Now you can add or delete fields in the template by using the template editor.

4.3.9 File Manager Base utility wizards

Many of the same utilities that are found by using the green screen version of File Manager can be found by using the File Manager plug-in. Wizards for more File Manager utilities are available by right-clicking in the Systems Information view and selecting the wanted utility. The utilities are described next.

Allocate Data Set

The Allocate function allows for the allocation of the following types of QSAM and VSAM data set types: PS, DA, PDS, LIBRARY, KSDS, RRDS, ESDS, and LDS.

By using the wizard, you can set the following basic options when a data set is allocated:

- System: Changes the system on which the function is run.
- Data Set: The name of the new data set to allocate.
- Allocate like (optional): The name of the data set to base all of the properties on for allocation of the new data set. Any fields that are specified later in the wizard override these properties. For convenience, the properties of the data set specified here are loaded into the relevant fields of the wizard.
- Data Set Type: The type of the new data set to allocate.
- Allocation Parameters (QSAM only).

The following utilities also are available:

- Compare

  By using the Compare utility, you can compare data from any resource to data in any other resource. You can compare load modules, perform a field level comparison (by using an Old copybook or template with a New copybook or template), and produce comparison reports. The reports show information about where insertions, deletions, or changes occurred in the New resource.

- Copy

  The Copy utility function is used to copy data from any supported resource to any other supported resource. By using the copy utility, you can copy selected members or all members from resources that contain members.

- Create Data

  Use the Create Data utility function to create and initialize data values in VSAM data sets, sequential data sets, PDS members, and HFS files.

  You specify the number of records that are to be created and how they are to be initialized. You can initialize the data with fill characters and patterns. When you use a copybook or a template, you can initialize records at the field level. You can change the field create attributes for individual fields by editing the copybook or template.

- Print

  The Print utility function is used to print data of any supported resource.

- Rename

  By using the rename wizard, you can rename data sets and members.

- When the rename wizard is open, enter a valid new fully qualified data set name for data sets. Entering the same name in a different case is invalid. However, if you enter a new data set name in lowercase, it automatically converts to uppercase during the rename operation. For members, enter only the member name. As with new data set names, new member names are upper-cased automatically during the rename operation. You must provide at least one valid new name before you can click **Finish**. If a new name is invalid, it is displayed in red with an error icon and an error message.
- View Load Module

  Use the View Load Module utility to print a list of the symbols (CSECTs, common sections, entry points, and ZAPs) in a load module. To compare load modules, use the Compare wizard.

4.3.10 File Manager DB2 utility functions

This section defines the following File Manager DB2 (FM/DB2) utility functions:

- Copy: Copies data from one DB2 object to another DB2 object.
- Create data: Loads a DB2 object with test data.
- Export: Exports data from a DB2 table or view to a QSAM or VSAM file.
- Import: Imports data, in a variety of formats, from a QSAM, or VSAM file, into a DB2 object.
- Print data: Prints the contents of a DB2 object in a variety of formats.

4.3.11 File Manager plug-in: resource navigation tips

Because the use of the File Manager plug-ins allows you to manipulate a variety of data that is stored on a z/OS system, you can identify and specify a resource name in different mechanisms, depending on your preferences.

Adding queries to the Systems Information view

The most basic method to add queries in the Systems Information view to browse and identify a resource to manipulate is to use the File Manager plug-in. As shown in Figure 4-59, all supported resource types are shown in the view. By using this view, you can browse to a resource in the following ways:

- Browse nodes (for example, Browse Data Sets, Browse DB2 Subsystems). This method is useful when you are unfamiliar with the z/OS environment on general or not familiar with the system to which you are currently connected. The browse nodes help you identify logical sets of resources that are available on your system. For example, if you expand Browse Data Sets node, it retrieves a list of data sets with your user name as their high-level qualifier.
- You can add queries to look up resources according to your criteria. For example, you can add query to look up resource which contains COPY in the name. Queries are persistent, so they are saved as part of the Systems Information view. After you successfully create a query, you can start a File Manager function against it from the view.

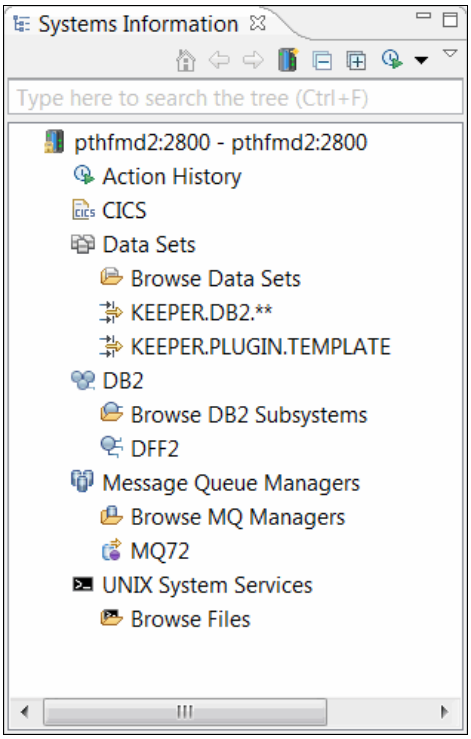


Figure 4-59 Adding queries to the Systems Information view

Lookup dialogs

The File Manager plug-in provides a lookup button throughout the product, which helps you to look up a resource name that is appropriate for the currently selected functionality. For example, Figure 4-60 shows a Lookup Resource window that helps you to look up a resource name to edit or view by using the formatted editor. Because the Lookup Resources window is started from the File Manager Base formatted editor options window, you can look up data sets, UNIX System Services files, CICS resources, and WebSphere MQ queues. Similarly, if you click a lookup button in the DB2 editor options window, you can look up only DB2 objects.

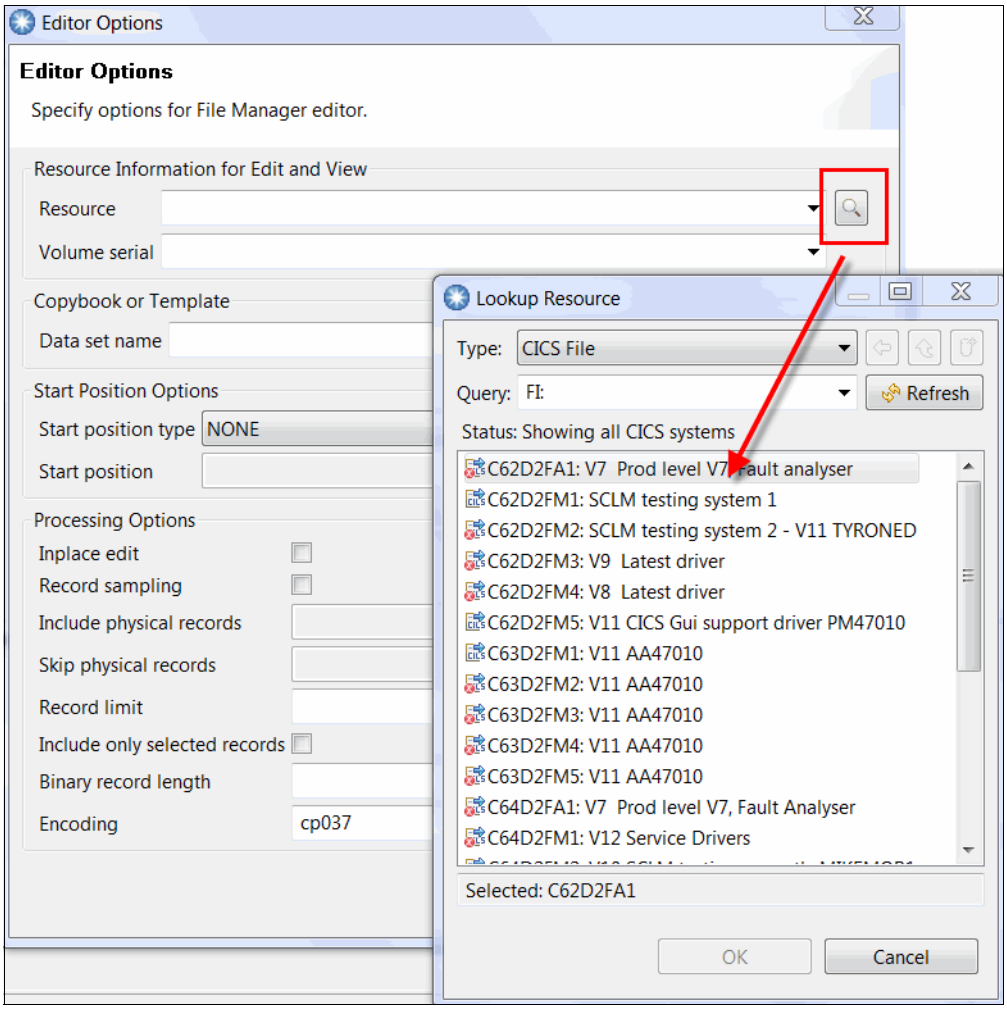


Figure 4-60 Look up Resources window from the File Manager Base editor options window

Content assist

Content assist is a convenient resource lookup mechanism that is provided by the File Manager plug-in. As you enter a prefix of a resource to manipulate, you press Ctrl+Spacebar to start the content assist functionality. Based on the entry location (the resource type that is allowed in the context) and the value that is specified, the File Manager plug-in looks up the resources that are appropriate for the current context.

Figure 4-61 shows an example of content assist usage in the formatted editor options window. The File Manager plug-in provides a list of data set names that start with the specified prefix (KEEPER.PLUGIN). You can enter more characters to narrow the result, or select a list, for example, Keeper.Plugin.People.KSDS, from the KEEPER.PLUGIN list. The information is looked up in real time as you press Ctrl+Spacebar.

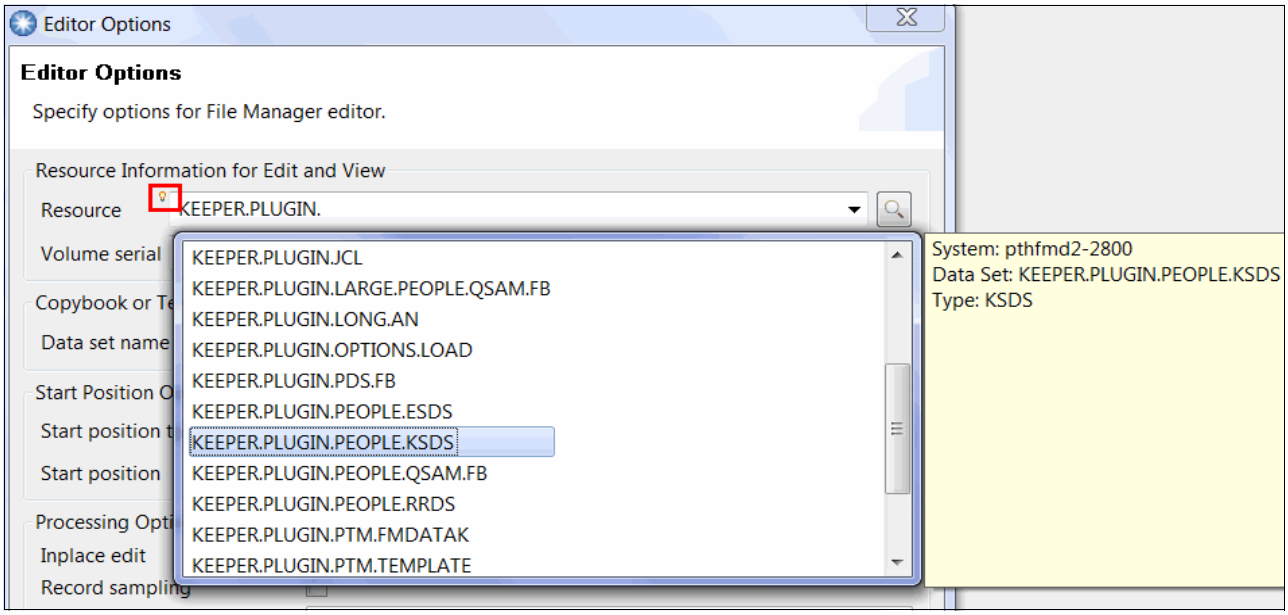


Figure 4-61 Content assist: Looking up data set names that start with the specified prefix

Direct input of a resource name

Most resource name entry fields in the File Manager plug-in support direct entry. If you are a power user who knows the exact name of a resource to manipulate, you can enter the resource name directly without using the resource navigation aid that is described in this section. This ability avoids unnecessary mouse clicks when they are not required.

4.4 Debug Tool

When IBM Debug Tool Plug-in for Eclipse is used with an Eclipse-based platform (such as, CICS Explorer, IMS Explorer, or Rational Developer for System z), it provides an interactive source debugging of z/OS applications. Debug Tool helps you debug applications that are running in various environments, such as Batch, Customer Information Control System (CICS), IMS, DB2 stored procedures, and UNIX System Services. It offers easy access through a GUI to the power of Debug Tool for z/OS. The GUI provides a workstation alternative to the z/OS ISPF interface, and can lower the learning curve for new z/OS developers. Along with making new developers proficient quicker, the GUI interface provides experienced users an easier and more flexible interface.

4.4.1 Introduction

By using the debugger, you control the execution of your program by setting breakpoints, suspending programs that were started, stepping through your code, and examining the contents of variables.

Host requirements

The Debug Tool Eclipse plug-in communicates with a Debug Tool z/OS session by using TCP/IP. Considerations for using the Debug Tool plug-in can be found at this website:

http://pic.dhe.ibm.com/infocenter/pdthelp/v1r1/index.jsp?topic=%2Fcom.ibm.debugtool.doc_11.1%2Feqabrs0110.htm

The graphical user interface

The information in this section expands on the information that is contained in the Getting Started portion of the readme file that is included with the plug-in.

The following steps that were used to start the Debug Tool Plug-in for Eclipse were compiled by using V12.1 of the code.

The Debug Tool plug-in, when integrated with Debug Tool for z/OS and the Eclipse-based platform, debugs z/OS-based load modules that are running on MVS, including load modules that run in a subsystem like CICS, DB2, or IMS.By using the GUI interface, you access a debugging capability similar to the capability that is accessed by the ISPF interface. By using the GUI interface, you can complete the following tasks:

- Set and clear breakpoints at a specific line.
- Set and clear breakpoints for an error or warning-level error that is based on Language Environment severities.
- Run to a breakpoint.
- Step into a procedure.
- Step over a procedure.
- View variable values and change them as you step through the code.
- View variable values in the context of a larger area of storage.
- View the call stack.

4.4.2 Opening the Debug Perspective

Complete the following steps to access the Debug Tool for z/OS GUI:

1. Click **Window** → **Open perspective** → **Other**, as show in Figure 4-62.

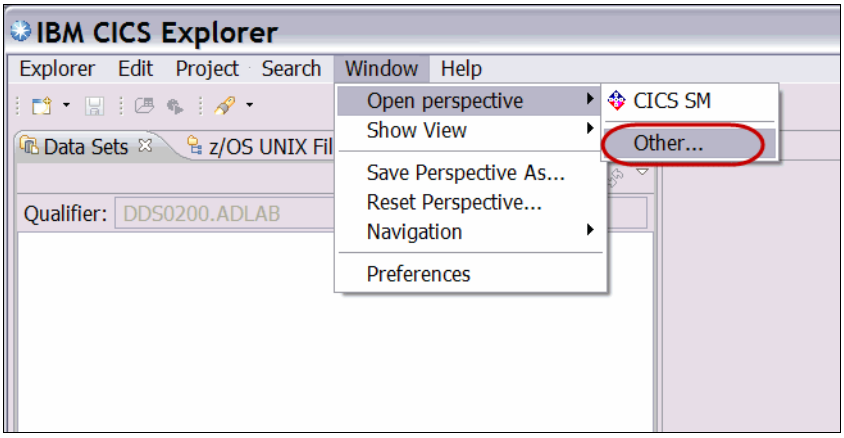


Figure 4-62 Open Perspective

2. Select **Debug Tool**, as shown in Figure 4-63.

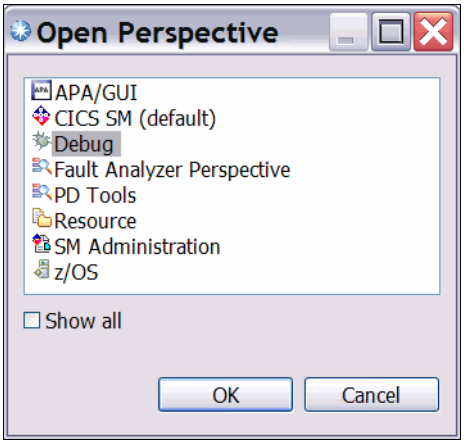


Figure 4-63 Select Debug

To change the default perspective to a perspective that you feel comfortable with, select **Window** → **Show View**. From the drop-down list, select the windows that you want to include in your default perspective. The Debug Console is the same as the Log window in a 3270 debug session. Save the changes as your default perspective by selecting **Window** → **Save Perspective As**.

4.4.3 Setup for debugging

To use an interactive debugger, we must tell the debugger which application must be debugged, and where to interact with us (as the user). To route an interactive debug session to the Debug GUI, ensure that the Debug Tool listener is listening for an interactive debug session and determine the IP address of the workstation where it is displayed.

These tasks can be completed by using the Debug Tool GUI listener button in the Debug Perspective. If the listener button is green, as shown in Figure 4-64, the Debug GUI is ready for an incoming debug session. If it is red, you can click the listener button to start the listener.

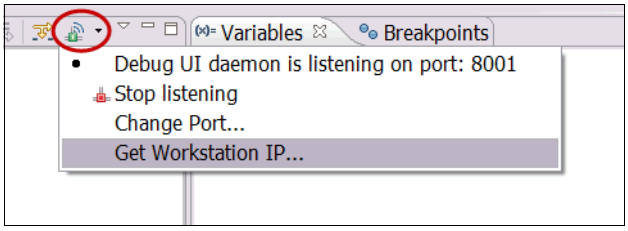


Figure 4-64 Debug Tool Listener button

Get the IP address by pulling down the connection detail in your Debug view and selecting **Get Workstation IP**. To communicate with your workstation, the debug session that is running on the host must know this IP address.

This function returns the IP address that is required by the Debug Tool server code to establish a connection with your workstation. If you are using a VPN client, two IP addresses are shown, as shown in Figure 4-65.

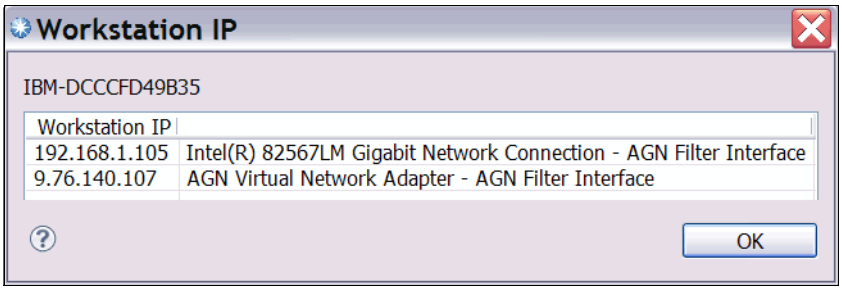


Figure 4-65 Workstation IP address

Alternatively, the address can be obtained by opening a Command Prompt window and entering the **`ipconfig`** command.

Debugging Batch Applications

Make note of the address that is returned and the port number where the Debug GUI is listening. Include this information in the TEST parameter in the JCL for the execution of the program you are about to debug. Modifying this TEST parameter in the JCL can be done from ISPF, the CICS Explorer z/OS perspective, or Rational Developer for System z, as shown in Figure 4-66.

```
//CEEOPTS DD *
  TEST(,,,TCPIP&9.76.140.107%8001:)
```

Figure 4-66 Debug GUI CEEOPTS DD statement

After the CEEOPTS DD statement with the appropriate TEST option is added to the runtime JCL, submit the batch job to start the debugging session.

Debugging CICS Applications

When a Customer Information Control System (CICS) program is debugged, the address that is returned and the associated port must be included in the 3270 DTCN profile, as shown in Figure 4-67.

```
Select type and ID of debug display device
 Session Type     ==> TCP                  MFI, TCP
 Port Number      ==> 8001                 TCP Port
 Display Id       ==> 9.76.140.107
```

Figure 4-67 3270 DTCN Profile builder

You also can use the DTCN profile builder that is installed with the Debug Tool GUI. This plug-in helps you create and manage DTCN profiles on your z/OS system from the Debug Perspective.

Complete the following steps to use the DTCN plug-in:

1. Verify that you completed all the tasks described earlier in this chapter.
2. Click **Window** → **Show View** → **Other**.
3. WIthin the Debug folder, select **DTCN Profiles** (as shown in Figure 4-68) and click **OK**.

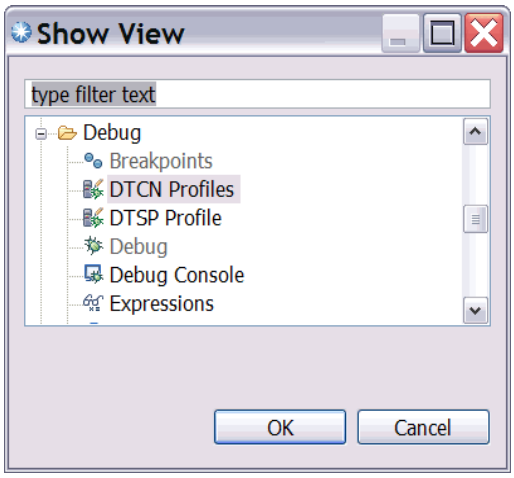


Figure 4-68 Show the DTCN view

The view is shown in the bottom portion of the perspective near the Console view. If you want to keep the DTCN view visible in the Debug Perspective, save the perspective.

4. Establish a connection to the DTCN profile manager by completing the following steps:

   a. Click **Window** → **Manage Connections.** Select **DTCN** and click **Add**, as shown in Figure 4-69.

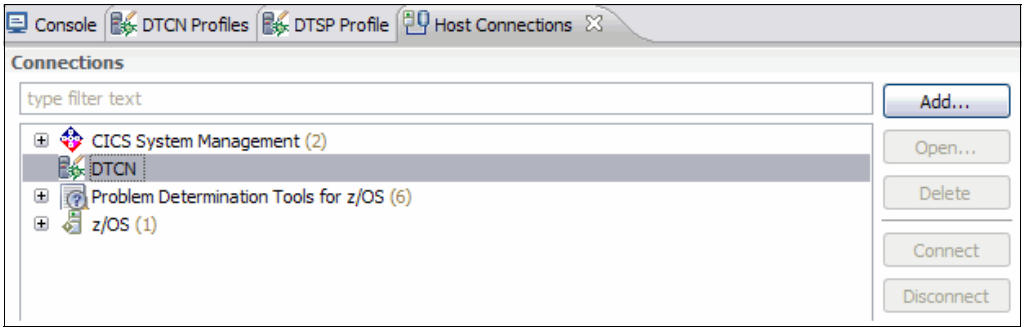


Figure 4-69 Host Connections

   b. Enter the host name and port number (which is supplied by the system administrator) for the CICS region that you are using for the debug session. The DTCN connection name is automatically generated as hostname:port after you enter the host name and port number details, as shown in Figure 4-70. Click **Save and Connect**.

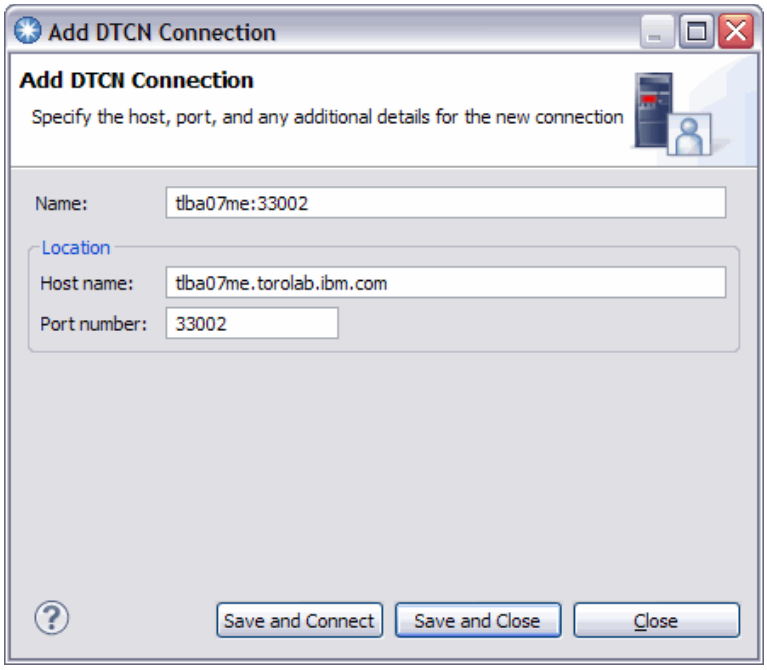


Figure 4-70 Add DTCN Connection

c. Complete the following steps that are based on the credentials that already are defined or are being created:

i. If no credentials are defined, enter the User ID and Password or Passphrase. The Credentials Name is automatically generated as UserId@hostname, as shown in Figure 4-71. Click **OK**.

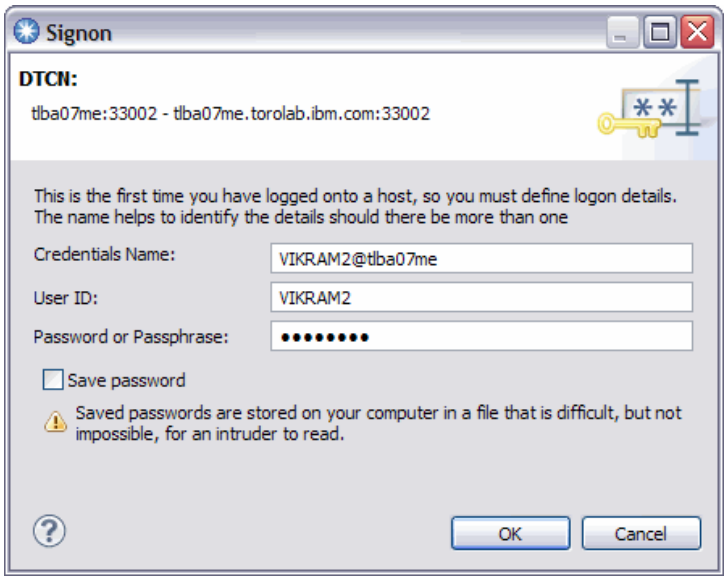


Figure 4-71 Sign on while the User ID and Password or Passphrase are specified

ii. If the credentials are already defined, select the credential that you want to use from the Credentials Name drop-down list, as shown in Figure 4-72. Click **OK**.

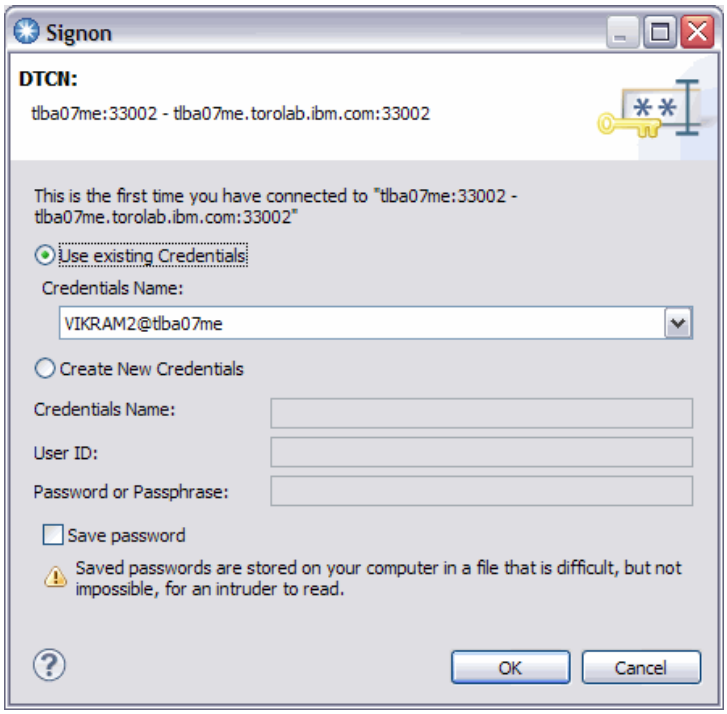


Figure 4-72 Sign on by selecting existing credentials

d. A successful DTCN connection is shown with a green icon, as shown in Figure 4-73.

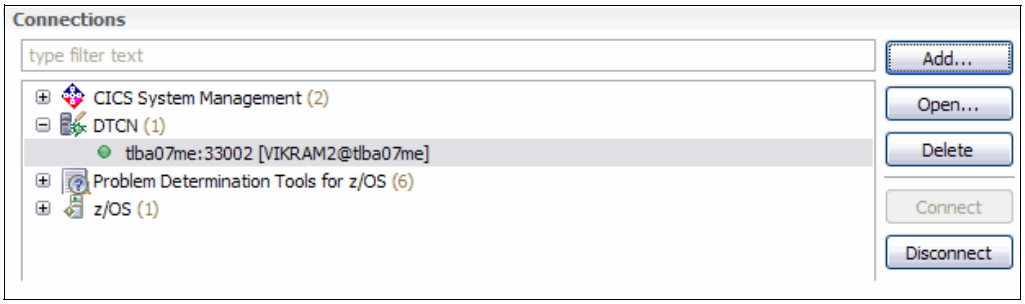


Figure 4-73 Successfully connected to DTCN server

5. In the DTCN Profiles view, which shows a list of profiles in the region you are working with, right-click anywhere to see a list of the actions that are available for working with profiles. Select **Create** to create a profile, or **Edit** to edit an existing profile, as shown in Figure 4-74.

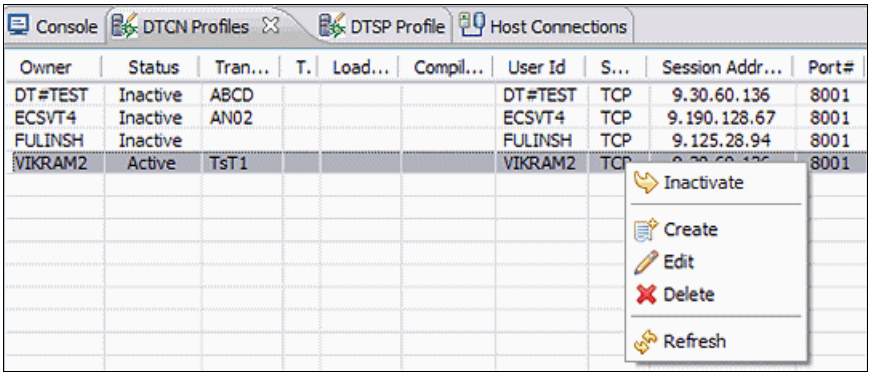


Figure 4-74 DTCN profiles view

a. In the DTCN Profiles view, double-click a DTCN profile to see the Debug Tool Profiles Management window, as shown in Figure 4-75. Click **Next**.

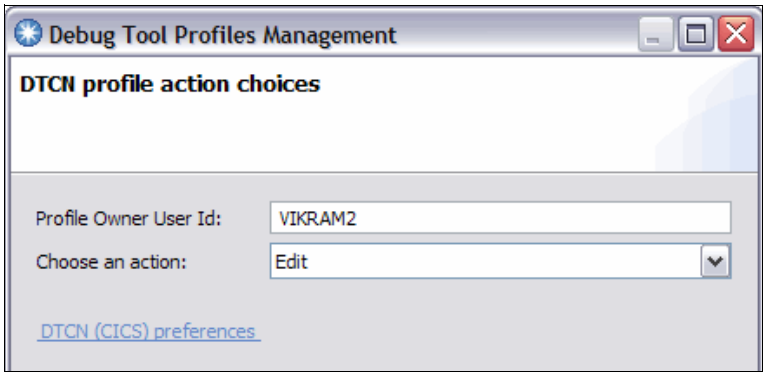


Figure 4-75 DTCN profile action choices

b. In the next window (see Figure 4-76), you modify the profile to capture the CICS application you want to debug, enter the required information and click **Next**.

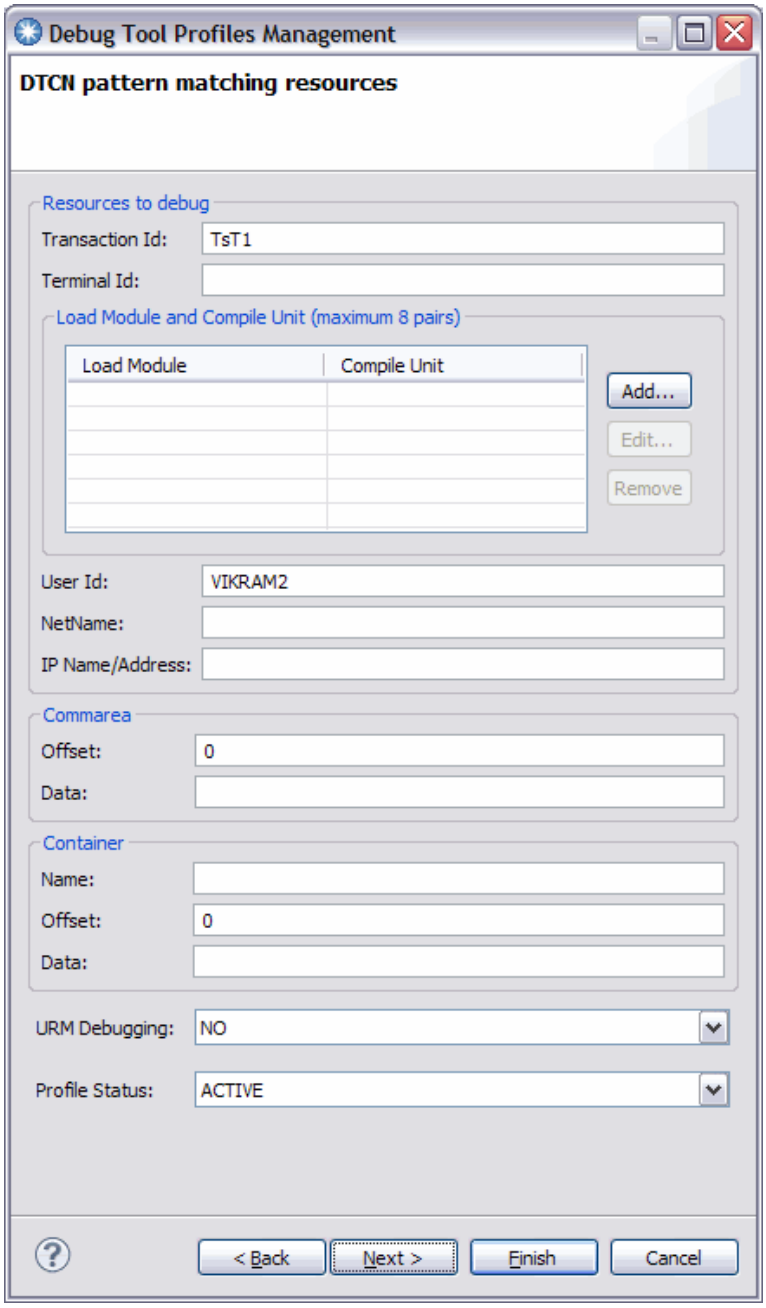


Figure 4-76 DTCN pattern matching resources window

c. In the final panel (see Figure 4-77), modify the location in which the debugging session is displayed. Selecting TCP for the session type updates the Session Address to the current TC/PIP address. Selecting TCP also displays the debugging session on the current workstation.

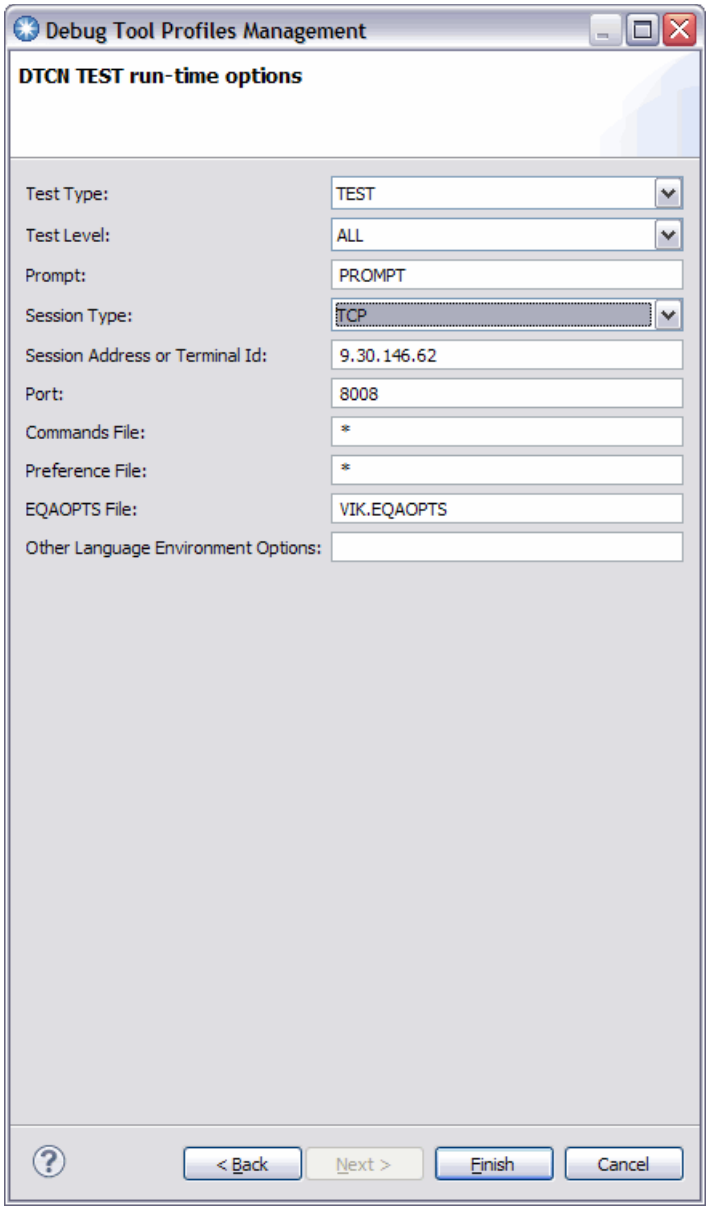


Figure 4-77 DTCN TEST runtime options

6. Click **Finish**. The profile is added to the DTCN Profiles View.

When the transaction is run again, the debug session begins.

4.4.4 Debugging COBOL, PL/I, Assembler, and C++ programs

After the job is submitted or the transaction that was set up for debugging is started, the debug session starts. You can now work in your debug session. You can step through your program, set a breakpoint, run your program to the breakpoint, and monitor variables in your application, as shown in Figure 4-78.

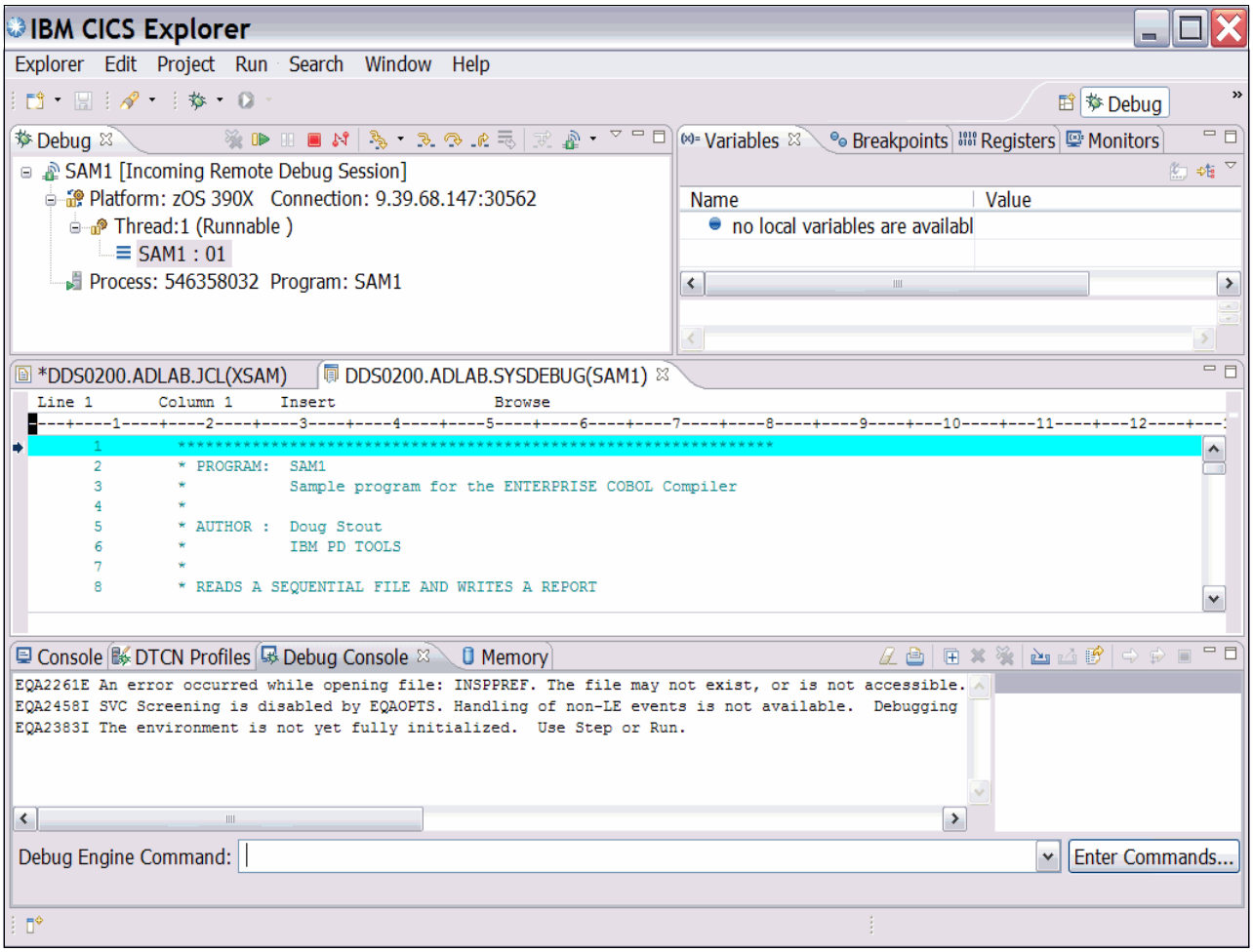


Figure 4-78 Debugging session on the Debug GUI

After a debug session is started, debug views provide access to various debug tasks. Views that are available for debugging are described in Table 4-2.

Table 4-2 Default Debug Perspective views

Views	Description
Debug View	Manages program debugging
Debugger Editor	Displays source for your program
Breakpoints View	Offers a convenient location for setting and working with breakpoints
Variables View	Contains a list of all variables in your application, which you use to edit variables
Registers View	Displays registers in your program
Monitors View	Provides a convenient location for working with variables, expressions, and registers that you choose to monitor
Modules View	Displays a list of modules that are loaded while your program is run
Debug Console	Issues commands to the debug engine, view output from the engine, and see results of commands that you issued
Memory View	View and map memory that are used by your application

Call stack

A call stack is a dynamic stack data structure that stores information about the active subroutines of a computer program. A call stack is often used for several related purposes. However, the main reason for having a call stack is to track the point to which each active subroutine must return control when it finishes running. When a program is suspended, the call stack looks similar to the call stack shown in Figure 4-79.

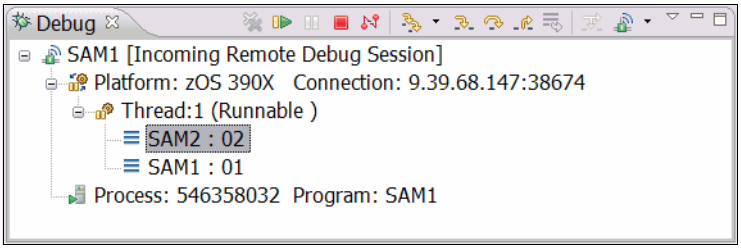


Figure 4-79 Call stack

Breakpoints

Breakpoints are temporary markers that you place in your executable program to tell the debugger to stop your program at a specific point. When a breakpoint is encountered, the program is suspended at the breakpoint before the statement is run. You can now see the stack for the thread and check the contents of variables, registers, and memory. You can then take a step (run) the statement and see what effect it has on the argument.

There are several methods for setting breakpoints. The Breakpoints view can be used to access the wizards for setting breakpoints by right-clicking in the view and selecting **Add Breakpoint** from the pop-up menu. A menu expands from which you choose the breakpoint type that you want to set, as shown in Figure 4-80.

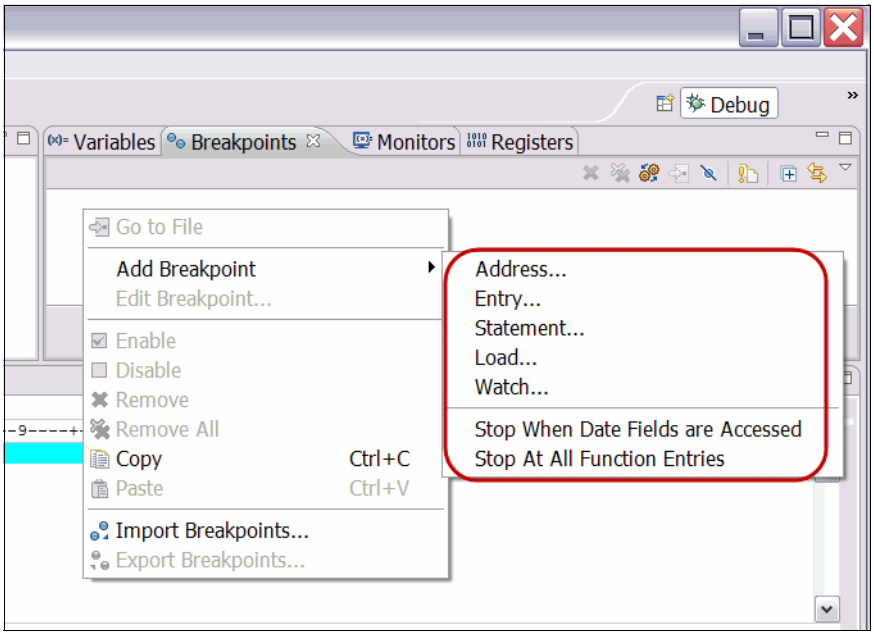


Figure 4-80 Add Breakpoint menu

The breakpoint types that are supported when debugging System z applications are listed in Table 4-3.

Table 4-3 Breakpoint descriptions

Breakpoint type	Description
Statement Breakpoint	Triggered when the statement it is set on is about to be run
Entry Breakpoint	Triggered when the entry points they apply to are entered
Address Breakpoint	Triggered before the disassembly instruction at a particular address is run
Watch Breakpoint	Triggered when execution changes data at a specific address
Load Breakpoint	Triggered when a DLL or object module is loaded
Exception Breakpoint	Triggered when an exception that is recognized by the debugger is thrown

By using the breakpoint wizard, you specify optional breakpoint parameters and set conditional breakpoints. In Figure 4-81, the condition of X = Y is set in the expression field, meaning that the breakpoint triggers only when the variable X equals Y. If an action is entered in the Action field, it takes place only if the breakpoint is triggered. The example shows moving 1 to variable X.

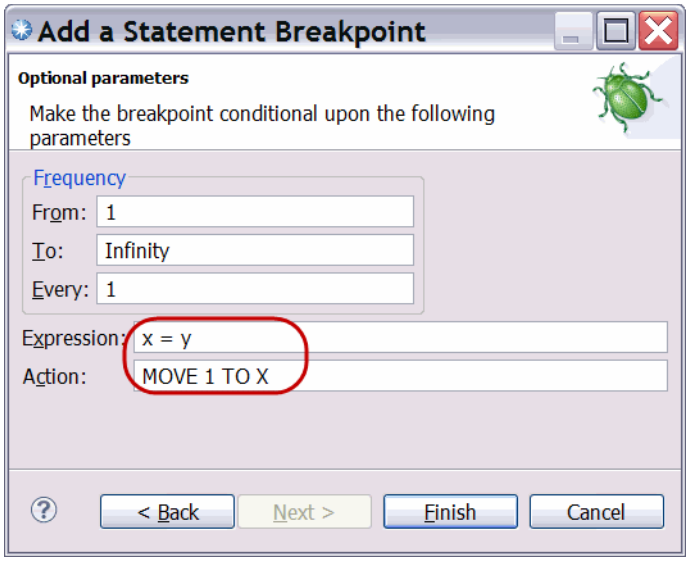


Figure 4-81 Breakpoint optional parameters

You can set exception breakpoints in the Breakpoints view by clicking **Manage Compiled Language Exception Breakpoints**. You select the exception type that you want the debugger to catch in the Manage Exception Breakpoints dialog box. Exception types that are available when debugging System z are TEST(ALL), TEST(ERROR), and TEST(NONE). You can select only one of these selection types.

You can set statement breakpoints in the Debugger Editor by double-clicking in the gray ruler area to the left of a statement or by right-clicking in the pop-up menu. Also, you can set the breakpoints by using the wizard in the Breakpoints view. Each statement breakpoint includes an indicator on the statement within the Source view, as shown in Figure 4-82.

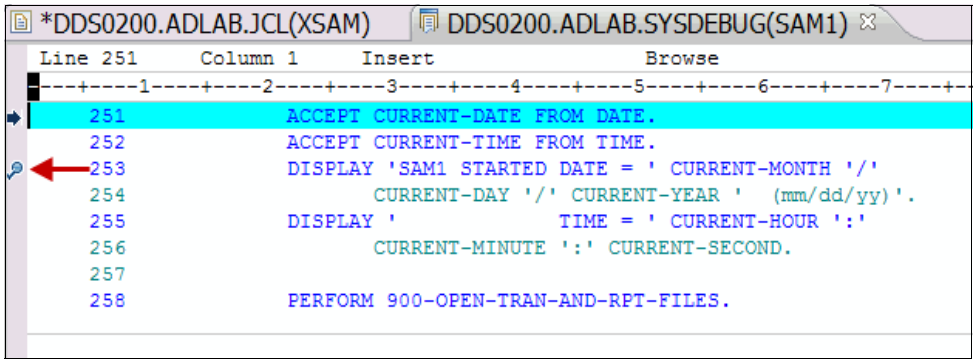


Figure 4-82 Statement Breakpoint indicator

You can set entry breakpoints in the Modules view by right-clicking an entry point and selecting **Set entry breakpoint** from the pop-up menu. You also can set the breakpoints by using the wizard in the Breakpoints view.

In addition, you can right-click the debug target (program name, one of the threads, or stack frames) in the Debug view. Select **Options** → **Stop At All Function Entries** from the pop-up menu to stop at all entry points (this option also is available in the Breakpoints view pop-up menu). All other breakpoint types are set by the use of the wizard in the Breakpoints view. A list of all breakpoints (for all debug sessions) is shown in the Breakpoints view, unless you use the filter by debug target action or link the Breakpoints view to the Debug view.

To filter breakpoints that are not related to the current debug session, click the Breakpoints view, then **Show Breakpoints Supported by Selected Target**.

To link the Breakpoints view with the Debug view, click **Link with Debug View**. When this toggle is selected and a breakpoint suspends a debug session, that breakpoint automatically is selected in the Breakpoints view.

The breakpoint entries in the list provide you, in brackets, with a summary of the properties of the breakpoints. The breakpoint that is shown in Figure 4-83 is a statement breakpoint that is set in program SAM1 on statement 253. By using the right-click pop-up menu options in the view, you add breakpoints, remove breakpoints, and enable or disable breakpoints. You can also edit breakpoint properties with a pop-up menu option. By using the push buttons in the Breakpoints view, you also can remove breakpoints.

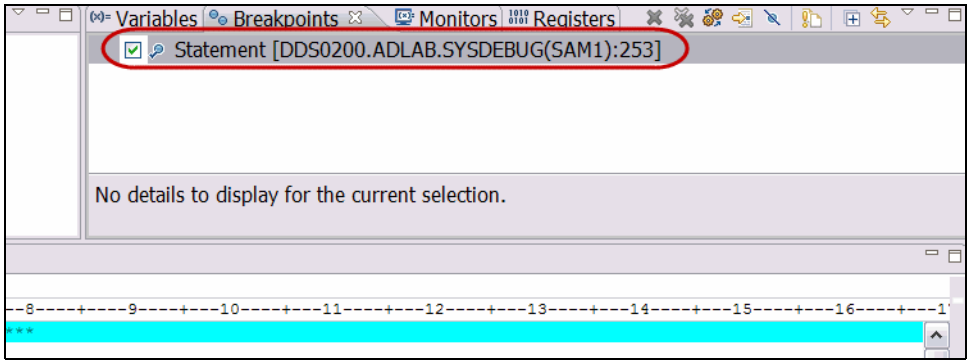


Figure 4-83 Breakpoints view

You can enable and disable breakpoints by using the pop-up menus in the Breakpoints view, the editor, or by selecting a check box in the Breakpoints view. When a breakpoint is enabled, it causes all threads to suspend whenever it is hit. When a breakpoint is disabled, it does not cause threads to suspend.

In the Breakpoints view, there are two indicators to the left of a set breakpoint. To the far left is a check box that indicates whether the breakpoint is enabled (when enabled, the check box contains a check mark). To the near left, an indicator with a check mark indicates a breakpoint that was successfully installed by the debug engine (if the breakpoint is enabled, the check box is selected; if the breakpoint is disabled, the check box ID cleared).

In the editor, statement breakpoints are indicated by a selected check box, indicating a breakpoint that was successfully installed by the debug engine (if the breakpoint is enabled, the check box is selected; if the breakpoint is disabled, the check box ID cleared).

Breakpoints must be installed before they suspend execution. It is possible to add a breakpoint that is invalid for the current debug session. This breakpoint is not installed until it is part of a debug session, which includes a debug engine that recognizes the breakpoint.

In the editor, statement and entry breakpoint indicators are displayed in the marker bar to the left of the editor. Indicators for statement, entry, address, watch, and load breakpoints are displayed in the Breakpoints view.

While in the Breakpoints view, you open the source editor to the location of a breakpoint by using one of the following methods:

- Double-click the breakpoint.
- Select the breakpoint and click **Go to File For Breakpoint**.
- Right-click the breakpoint and select **Go to File**.

Stepping procedures

After the code is suspended, you use the stepping procedures to step through the execution of the program line-by-line by using the step controls, as shown in Figure 4-84.

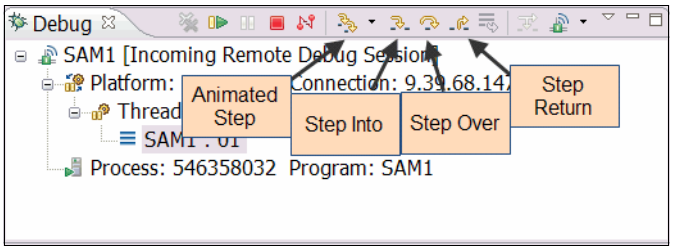


Figure 4-84 Stepping procedures

While a step operation is performed, if a breakpoint or exception is encountered, execution suspends at the breakpoint or exception, and the step operation ends. You can use step commands to step through your program a single instruction or location at a time. The available step commands are described in Table 4-4.

Table 4-4 Step Actions

Command	Description
Step Over	The called functions are run without stepping into them.
Step Into	The program runs to the next hook or statement.
Animated Step Into	The debugger issues a **`Step Into`** action repeatedly. You can control the delay between each step by selecting the Animated Step Into icon down-arrow.
Step Return	The program runs to the return point (just after the call point).

Select the **Step Over** command to step over the next method call (without entering it) at the currently running line of code. Although the method is skipped, it is still run.

Variable tools

Any function or program runs to change data in an expected way, so a key part of debugging is to locate and determine whether the program is acting on the data correctly. The first step is finding a variable. Locating and determining its value in a complex program while debugging can be tedious. The Debug Tool GUI features solutions for this task. In one method, when a breakpoint is set and the program is suspended, you can use the Hover feature. By using this feature, you see a variable's current value by placing the mouse pointer over the variable name while the program is suspended, as shown in Figure 4-85.

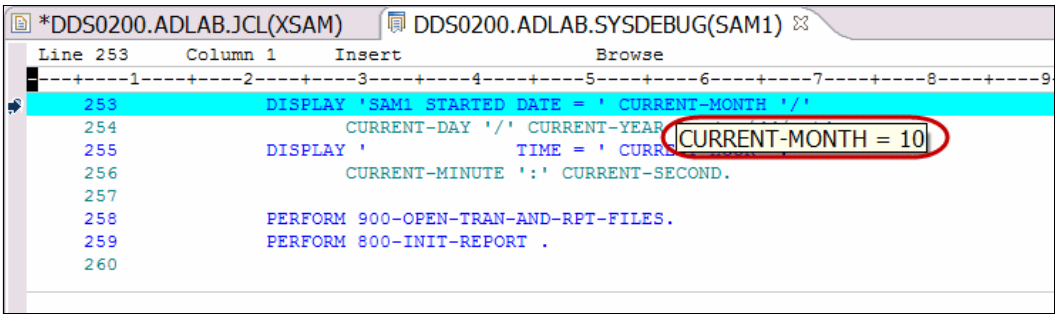


Figure 4-85 Hover feature

You can set up a window that shows the variable name and its value as you step through the program. To open this variables window, go to Window in the top toolbar, select **Show View**, then, near the bottom, you see the Variables tab. You also can press ALT+SHIFT+ Q, V, and a window is shown near your Breakpoints window, as shown in Figure 4-86.

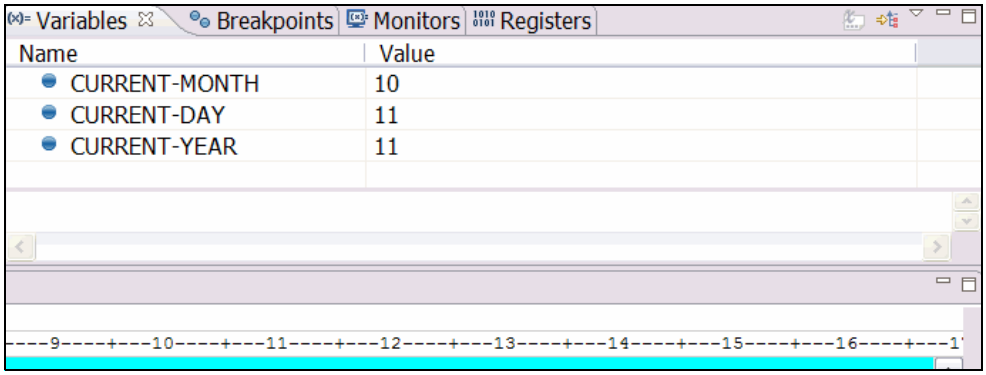


Figure 4-86 Variables View

In the Variables View, you can change the variable values as you step through the code. By using this feature, you enter the correct data into the variable for testing the rest of the program.

Debug Console commands

In the Debug Console View, place the cursor in the Debug Engine Command's text box. Press Ctrl+Spacebar to see a list of the Debug Tool commands, as shown in Figure 4-87.

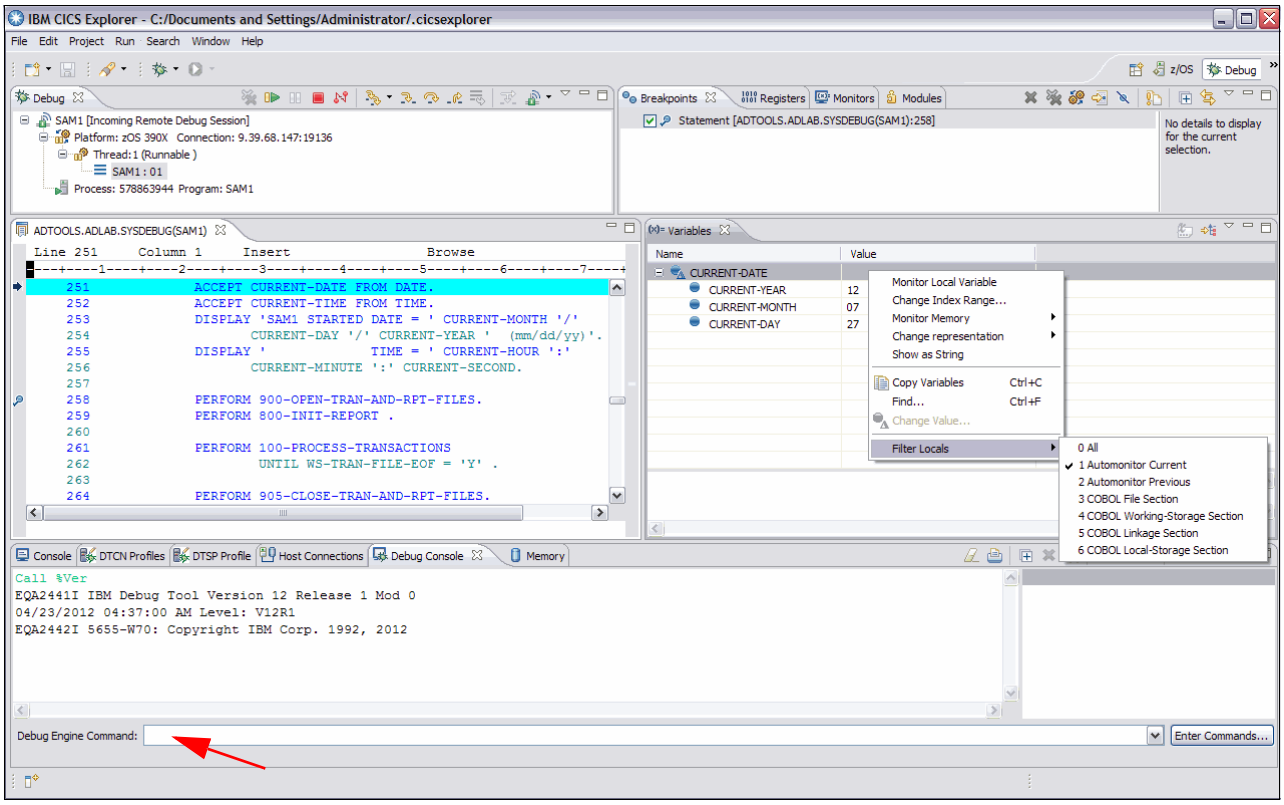


Figure 4-87 Press Ctrl+Spacebar for a list of Debug Tool commands

4.4.5 Right-clicking in the code space

During a debug session, You access many features by using right-click. Right-clicking in the code space during a debug session gives access to functionality (see Table 4-5) that is useful in debugging source code.

Table 4-5 Right-click menu in code space

Action Name	Description
Find Text	Enter a string of characters to find within the code.
Find Function or Entry Point	Enter the name of the function or entry point for which you are searching.
Add Watch Breakpoint	Set up a Watch Breakpoint by using the wizard. Set up the number of bites to watch and the frequency of the repetitions.
Monitor Expression	Select a thread that has the expression you want to monitor. The tool places a monitor in the monitor tab.
Monitor Memory	Add a memory monitor with which you view and change the contents of memory or memory areas that are used by your program.
Run to Location	Runs the program to the currently selected line in the editor.
Jump To location	Jumps to the currently selected line in the editor without execution.
Edit Source Lookup Path	Edits the path that is used to locate source files.
Change Text Files	Enters the name of an overriding file for the source.
Switch Views	Changes the views.

4.5 Fault Analyzer

The features of the Fault Analyzer are described in this section.

4.5.1 Fault Analyzer plug-in for Eclipse

In this section, we describe topics that are related to the Fault Analyzer plug-in for Eclipse.

Host requirements

The Fault Analyzer plug-in for Eclipse communicates with a Problem Determination Tools Common Server. You must ensure that the Fault Analyzer extension is configured on the server. For more information about installing the required server code, see the *IBM Fault Analyzer for z/OS User's Guide and Reference*, SC19-3131-05.

The graphical user interface

The Fault Analyzer plug-in for Eclipse simplifies abend analysis by connecting to a remote IBM z/OS system. It also provides access to problem reports for diagnosing mainframe application errors and abends. The Fault Analyzer plug-in for Eclipse include the following key features:

- An interface to manage views and multiple fault history files from any number of z/OS hosts
- The ability to list the contents of history files and views, and to configure column layouts and results filtering
- The ability to browse fault entries that were created during real-time analysis of abending programs
- A browser for browsing the dump storage that is associated with a fault entry and annotating areas of interest in the dump by using user notes
- A source listing of abending programs that are using side files
- The ability to look up message explanations for any abend or message code that is referred to by the report

4.5.2 Fault Analyzer Artifacts view

By using the Fault Analyzer Objects view (as shown in Figure 4-88), you register a set of history files that can be browsed to identify the cause of failures in your applications. You also can monitor the contents of history files to alert you when a failure occurs in one of your applications.

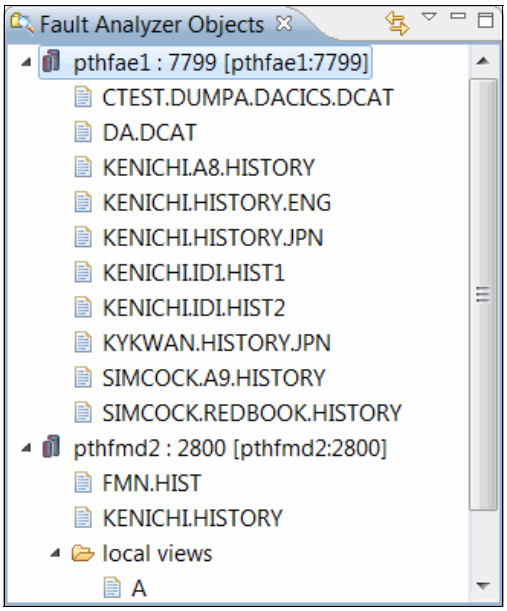


Figure 4-88 Fault Analyzer Objects view

History files are organized in a tree structure in which the root of the tree is the name of connection that is registered in the Host Connections view. You can add History Files or Views of your interest and create a local view, which is a collection of history files on your system.

When you double-click a history file element, all fault entries that are contained in the selected history file are displayed in the Detailed View. A view is a Fault Analyzer concept with which you group a set of history files. When you select a view element, fault entries that are contained in all history files that are defined in the selected view are displayed in the Detailed View.

To specify a history file in the Fault Analyzer Objects view, right-click in the view and select **Add History File** from the context menu, as shown in Figure 4-89.

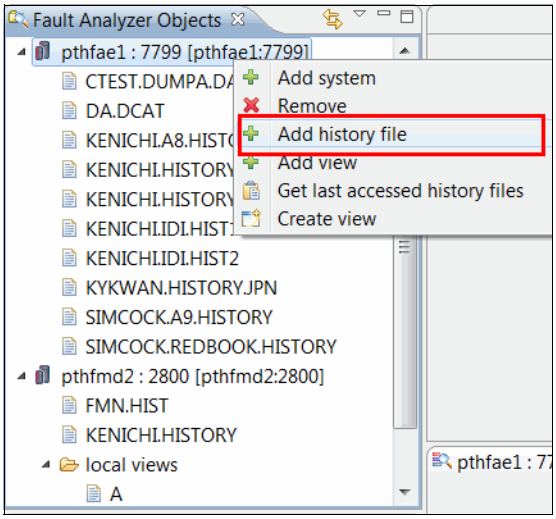


Figure 4-89 Adding a history file to the Fault Analyzer Objects view

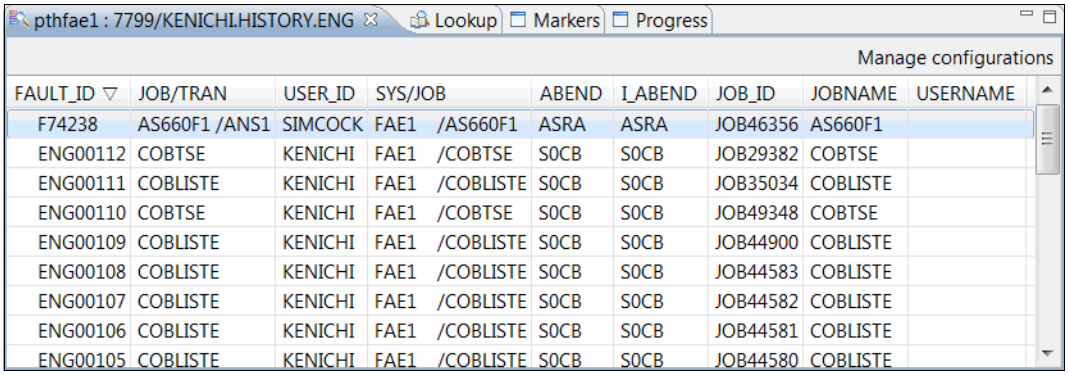
Table 4-6 provides a summary of the actions that are available in the Fault Analyzer Artifacts view.

Table 4-6 FA Artifacts View actions

Action Name	Description
Add system	Adds a connection to the view. You can also define a new connection to a PD Tools common server if you did not establish a connection.
Add history file	Adds a history file from a particular host to the view.
Retrieve last accessed history file	Retrieves the set of history files that you last used on the host, which is based on the information that is stored in your ISPF profile.
Retrieve view information	Retrieves information about views, which are stored in a data set.
Remove	Deletes the selected item from the view.
Refresh	Refreshes the cached information about the selected history file.

4.5.3 Fault Analyzer Detailed view

The Detailed view (as shown in Figure 4-90) displays the summary of fault entries that are contained in the selected history file or view in the Fault Analyzer Objects view. This view gives you a quick summary of what is happening on your system (a history file-centric view of the system). The column headings are configurable, depending on your preference and area of interest.



FAULT_ID	JOB/TRAN	USER_ID	SYS/JOB	ABEND	I_ABEND	JOB_ID	JOBNAME	USERNAME
F74238	AS660F1 /ANS1	SIMCOCK	FAE1 /AS660F1	ASRA	ASRA	JOB46356	AS660F1	
ENG00112	COBTSE	KENICHI	FAE1 /COBTSE	S0CB	S0CB	JOB29382	COBTSE	
ENG00111	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB35034	COBLISTE	
ENG00110	COBTSE	KENICHI	FAE1 /COBTSE	S0CB	S0CB	JOB49348	COBTSE	
ENG00109	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB44900	COBLISTE	
ENG00108	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB44583	COBLISTE	
ENG00107	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB44582	COBLISTE	
ENG00106	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB44581	COBLISTE	
ENG00105	COBLISTE	KENICHI	FAE1 /COBLISTE	S0CB	S0CB	JOB44580	COBLISTE	

Figure 4-90 Fault Analyzer view that displays the contents of a selected history file/view

From this view, you can start the Report browser (see 4.5.5, “Fault Analyzer Report browser”). By using the browser, you can see the detailed analysis of the selected fault entry or run the Dump browser (see 4.5.6, “Fault Analyzer Dump browser”) to browse a hex dump display of mini-dump pages that are stored in the selected fault entry.

Table 4-7 shows a summary of the actions available from this view.

Table 4-7 FA Report browser action

Action Name	Description
Column Configuration	Opens the column configuration dialog to manage named configurations
Columns	Enables fast selection of visible columns.
Filters	Applies wildcard filters to any column.
Open	Retrieves the report that is associated with the selected fault entry and displays the report. The report is cached locally after it is retrieved. The same effect can be achieved by double-clicking a fault entry.
Set options/Refresh	Opens the dialog to supply sidefiles (see Figure 4-95 on page 150).
Refresh Fault Entry	Refreshes the fault entry.
Clear cached data	Clears the cached information for the selected fault entry.

4.5.4 Working with older fault entries

For the plug-in to show fault entries in a history file correctly, you create the fault entries with the correct version of the Fault Analyzer feature and specify the appropriate set of options.

Whenever the plug-in detects a fault entry that does not contain information that is required for correct operation, it offers an option to refresh the fault entry, as shown in Figure 4-91.

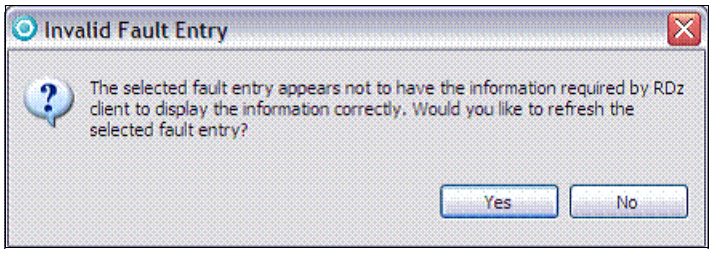


Figure 4-91 Invalid Fault Entry window

You can analyze fault entries to re-create the contents so that it can be viewed by using the client. If you refresh the fault entry, the plug-in's cached information that is associated with the fault entry is deleted. The fault entry is re-created, and the contents of the fault entry are downloaded to the client.

4.5.5 Fault Analyzer Report browser

The Fault Analyzer Report browser, as shown in Figure 4-92, is a multi-tabbed browser that displays the report that is associated with the selected fault entry. The browser is started by double-clicking a fault entry in the detailed view.

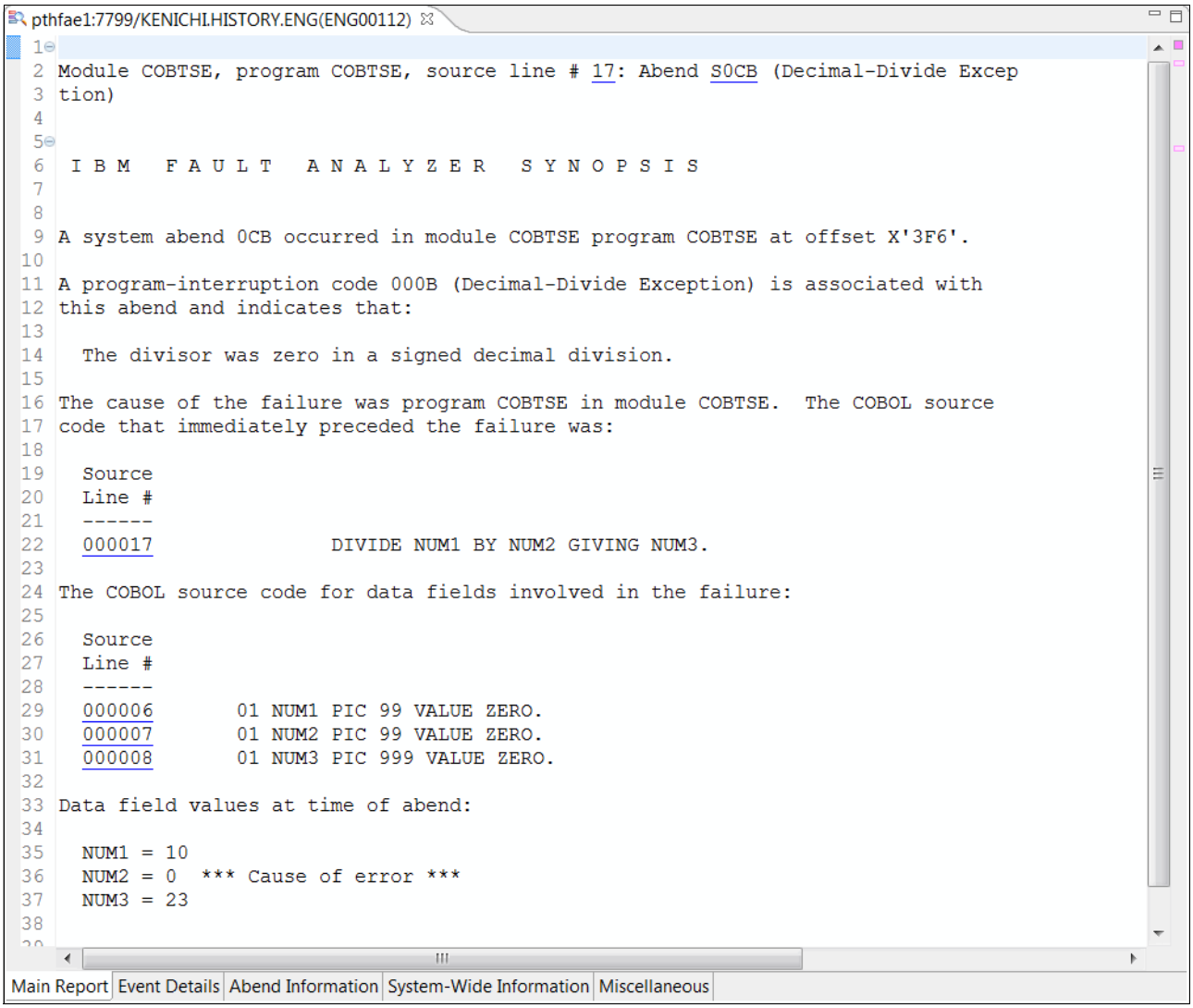


Figure 4-92 Fault Analyzer report that shows the result of abend analysis

Separate parts of a report are organized by using tabs for categories that show along the bottom of the window. The tabs separate the report into the following categories:

- Main Report

  This section includes the following sections:

  - Fault summary section

    A brief description of the fault, which includes the hyperlinked source line number of the abending program (if possible) and the abend code. If you click the source line number, the source code of the abending program where the failure occurred (see 4.5.7, “Source code display” for further information) is shown. If you click the abend code, the description (see 4.5.8, “Lookup view”) is shown.

  - Synopsis section

    The synopsis section provides a brief description of the fault and its analysis.

- Event Summary

  The Event Summary is a list of all events, in chronological order.

  The Event Details subsection provides detailed information about each event. More information that is associated with the event, such as message description and the contents of the program's working storage, are included in this subsection. The source code information or failing machine instruction is also included here.

- Abend Job Information

  This section provides the following information about the abending job that is associated with the real-time invocation of the Fault Analyzer feature:

  - Abend date
  - Time
  - Job ID
  - Job name
  - Job step name
  - Execution environment
  - Language Environment runtime options

- System-wide Information

  This section contains various types of information, including console messages that are not identified as belonging to any specific event or CICS system-related information, such as trace data and 3270 window buffer contents. Information about open files that cannot be associated with any specific event also might be included here.

  If there is no information in this section, then it does not show in the report.

- Miscellaneous Information

  This section includes the following types of information:

  - Fault Analyzer options: Provides lists of the Fault Analyzer options that were in effect at the time of the analysis.
  - Prolog section: Provides information about the version, release, and modification level of the Fault Analyzer, and the latest authorized program analysis report (APAR) or program temporary fix (PTF) that is installed.
  - Epilog section: Provides information about the invocation exists that are used. It also includes the approximate amount of above-the-line storage that is allocated during the analysis and the fault ID that is assigned. The time and date when the report was created also is included.

For a comprehensive description of the report sections, see the *IBM Fault Analyzer for z/OS User's Guide and Reference*, SC19-3131-05.

For easy navigation, an outline view is present in the lower left corner of the Fault Analyzer perspective, as shown in Figure 4-93.

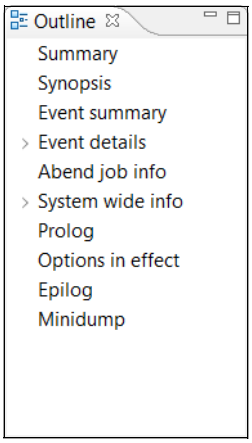


Figure 4-93 Outline of the currently selected report for easier navigation

As of V10.1.0.2 of the Fault Analyzer plug-in for Eclipse, the report browser was improved with the following features:

- Faster loading by using the Eclipse text editor framework for display and folding of large sections
- Highlighted section headings in the report, and in the overview ruler on the far right of the browser
- The ability to use markers to bookmark positions in a report, which are viewable through the markers view and the overview ruler
- Find and search capabilities for finding text in a report, or searching for text across multiple reports
- The choice to use the default sidefiles a report was created with to view the source, thus eliminating the need to provide sidefiles every time
- Dynamic highlighting of possible dump addresses in the report on mouseover

For more information about these features, see the plug-in's documentation by clicking **Help** → **Help Contents**.

4.5.6 Fault Analyzer Dump browser

The Fault Analyzer Dump browser (as shown in Figure 4-94) is a browser with which you browse the mini-dump pages that are stored for the selected fault entry in a history file. This browser often is opened by selecting the **Minidump** entry in the Outline view.

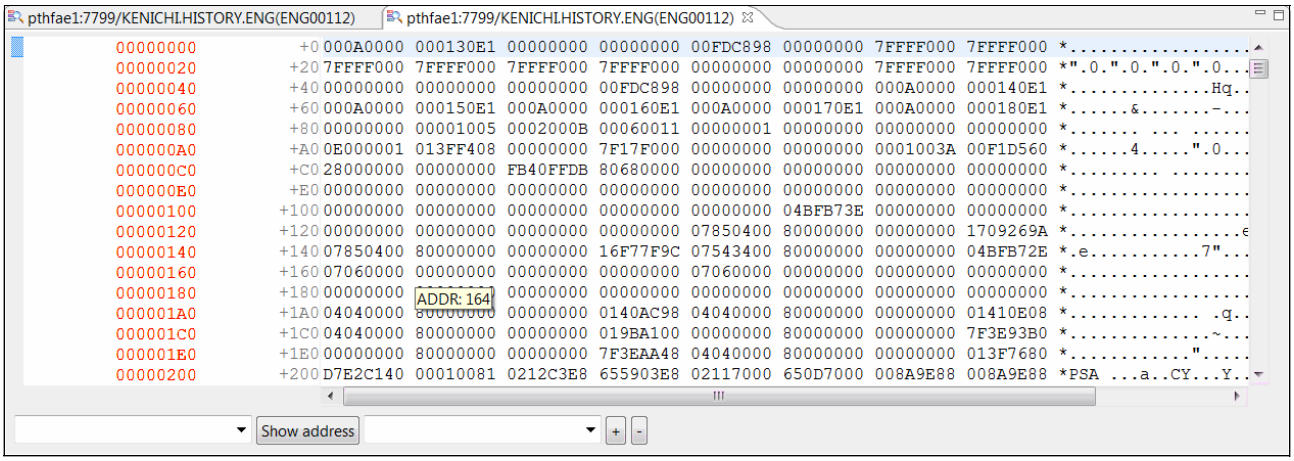


Figure 4-94 Mini-dump browser

4.5.7 Source code display

Fault Analyzer supports several different side-file formats to display the associated source code information of an abending program.

The Fault Analyzer plug-in for Eclipse supports the source line display of the abending program if the correct sidefile was available during the analysis. If the correct sidefile was available during the analysis of your abending program, the source line number of the program is hyperlinked in the report. When you click the source line number, it retrieves the sidefile from the host and generates the source line for display. The source file opens in the powerful LPEX editor in read-only mode. You have an opportunity to specify the data set names that contain your sidefiles during the fault entry refresh process (see Figure 4-95 and Table 4-8 on page 151).

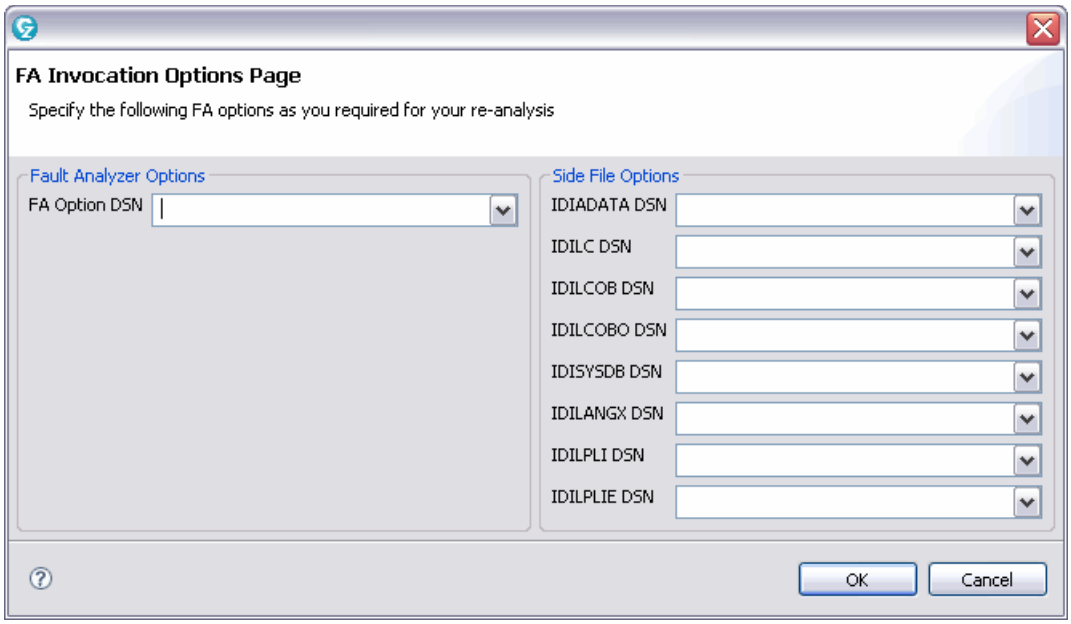


Figure 4-95 FA invocation options page

Table 4-8 FA invocation option descriptions

Action name	Description
FA Option DSN	A fully qualified data set name that contains the Fault Analyzer options (usually SYS1.PARMLIB(IDICNF00)). For more information, see the *IBM Fault Analyzer for z/OS User's Guide and Reference*, SC19-3131-05.
IDIADATA DSN	The name of one or more sequential or PDS(E) data sets that are holding Assembler SYSADATA files.
IDILC DSN	The name of one or more sequential or PDS(E) data sets that are holding C compiler listings.
IDILCOB DSN	The name of one or more sequential or PDS(E) data sets that are holding COBOL compiler listings (other than OS/VS COBOL).
IDILCOBO DSN	The name of one or more sequential or PDS(E) data sets that are holding OS/VS COBOL compiler listings.
IDISYSDB DSN	The name of one or more sequential or PDS(E) data sets that contain COBOL or Enterprise PL/I SYSDEBUG sidefiles. These side files are created when a COBOL program with the TEST(SEPARATE) options is compiled.
IDILANGX DSN	The name of one or more sequential or PDSE(E) data sets that are holding side files.
IDILPLI DSN	The name of one or more sequential or PDS(E) data sets that are holding PL/I compiler listings (other than Enterprise PL/I).
IDILPLIE DSN	The name of one or more sequential or PDS(E) data sets that are holding Enterprise PL/I compiler listings.

4.5.8 Lookup view

In the Lookup view, you can browse to the description of the abend codes, messages, and other miscellaneous information. You can browse the information by using one of the following methods:

- Find the description for a particular abend code, message, or miscellaneous information by navigating through the tree structure.
- Enter a pattern name in the Search box. All matching abend codes, messages, and miscellaneous information are displayed in the Results tab in the view. From the list of matching results, you can browse to find the description.

An abend code that is associated with a particular fault entry is hyperlinked in the report view. Click the hyperlinked abend code to show the description.

5

Using IBM Problem Determination Tools with CICS

In this chapter, we present four scenarios with Fault Analyzer, File Manager, Debug Tool, and Application Performance Analyzer. These scenarios illustrate how to find and fix various Customer Information Control System (CICS) transaction problems, such as abends and loops, by using the IBM Problem Determination Tools.

Where applicable, we describe the use of the tools from a host CICS terminal or a TSO/ISPF session, and from CICS Explorer on a workstation.

A CICS transaction that is named RED1 is the basis for all of the scenarios. When run, it displays a menu of tests, as shown in Figure 5-1.

```
                Redbook Test cases


_       Test Number 1  -  Open File error
_       Test Number 2  -  Bad Data
_       Test Number 3  -  Loop

Select test number above and press ENTER
```

Figure 5-1 RED1 main menu

To run a particular test, enter S next to the test and press Enter.

5.1 Using Fault Analyzer with CICS

The test program that we use in this scenario consists of a COBOL program (RDBKC01), which calls a PL/I program (RDBKP01) that abends. The test is run from the RED1 transaction menu that is shown in Figure 5-1 on page 153 by selecting Test Number 1 - Open File error.

Figure 5-2 shows the CICS transaction that is abnormally terminated with abend code AEXL.

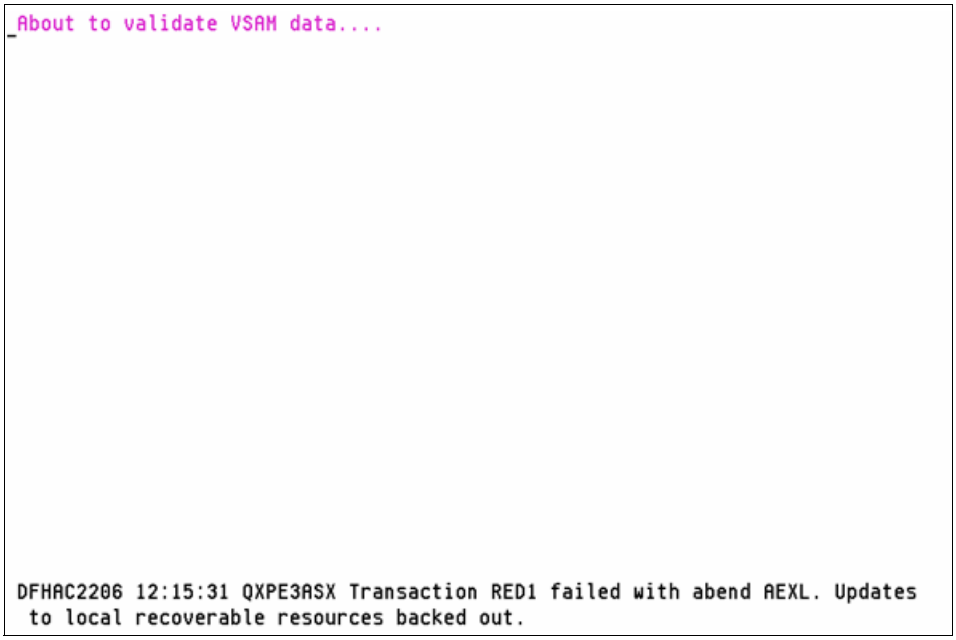


Figure 5-2 Transaction abend

Because Fault Analyzer was installed and enabled to capture CICS transaction abends on this system, a fault entry was written to the history file that was used by this CICS region.

5.1.1 Starting Fault Analyzer from a CICS terminal

As an alternative to using Fault Analyzer under TSO/ISPF, we used the following procedure to start Fault Analyzer directly from our CICS terminal to determine the reason for the abend:

1. Clear the window, enter `IFA`, and press Enter.

   The initial Fault Analyzer display shows the history file or view that was last used.

2. If this history file or view is incorrect, then specify another history file or view in the Fault History File or View field and press Enter to show the fault entries.

The Fault Entry List is displayed, as shown in Figure 5-3.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===> _____________________________________________  Scroll ===> CSR

Fault History File or View  : 'REDBOOK.HIST'_____________________________

{The following line commands are available: ? (Query), V or S (View saved
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history), C (Copy fault entry), M (Move fault entry), X (XMIT fault
entry).}

   Fault_ID Job/Tran User_ID  Sys/Job  Abend  Date       Time
_  F00013 RED1     SIMCOCK  AS650F3  AEXL   2010/06/04 12:15:26
_  F00012 RED1     SIMCOCK  AS650F3  ATSP   2010/06/03 20:59:45
_  F00011 CEMT     SIMCOCK  AS650F3  ATNI   2010/06/03 16:43:21
_  F00010 RED1     SIMCOCK  AS650F3  AEIT   2010/06/03 14:48:27
_  F00009 REDL     SIMCOCK  AS650F3  AEIR   2010/06/03 10:00:45
_  F00008 CEBR     SIMCOCK  AS650F3  ATNI   2010/06/02 15:30:44
_  F00007 RED1     SIMCOCK  AS650F3  ATSC   2010/06/02 14:46:33
_  F00006 RED1     SIMCOCK  AS650F3  ASRA   2010/06/02 08:32:39
_  F00005 RED1     SIMCOCK  AS650F3  ASRA   2010/06/01 16:13:50
_  F00004 RED1     SIMCOCK  AS650F3  ASRA   2010/06/01 11:28:55
_  F00003 RED1     SIMCOCK  AS650F3  AEXL   2010/06/01 11:09:14
```

Figure 5-3 Fault Entry List

The fault entry that was created for our abend is F00013. This fault entry is shown at the top of the window because no other faults were recorded in this history file since our transaction abended.

To see the Fault Analyzer report of our problem, we can use the **V** or the **I** line command. These commands are two methods that are used to show the same information, each with its own advantages. For our scenario, we choose the **I** line command.

3. Enter **I** next to the F00013 fault entry.

   The Interactive Reanalysis Report is displayed, as shown in Figure 5-4.

```
  File  View  Services  Help
Interactive Reanalysis Report                                  Line 1 Col 1 80
Command ===> _____________________________________________  Scroll ===> CSR
TRANID: RED1      CICS ABEND: AEXL              FAE3      2010/06/04  12:15:26

Fault Summary:
Module RDBKP01, program RDBKP01, source line # 22 : CICS abend AEXL .

Select one of the following options to access further fault information:
  1. Synopsis
  2. Event Summary
  3. CICS Information
  4. Storage Areas
  5. Language Environment Heap Analysis
  6. Abend Job Information
  7. Fault Analyzer Options

{Fault Analyzer maximum storage allocated: 4.22 megabytes.}
{DeferredReport processing execution time was 4.01 seconds (1.22 seconds CPU)}

*** Bottom of data.
```

Figure 5-4 Interactive Reanalysis Report

We can display the abending source line directly from the point-and-shoot field **22**. In Figure 5-4 on page 155, this field is outlined in red (it is normally shown in yellow).

However, in the Fault Summary, we can select menu option 2, which is Event Summary, to see more details about the abend. This option is available by either of the following methods:

- Enter **2** on the command line.
- Move the cursor to the **2** option, then press Enter.

The Event Summary is displayed, as shown in Figure 5-5.

```
  File  View  Services  Help
Event Summary                                                Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: RED1      CICS ABEND: AEXL              FAE3      2010/06/04  12:15:26

{The following events are presented in chronological order.}

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*)      Loaded
 1 Call               DFHAPLI  DFHAPLI1 n/a      P+2E6E                  CICS.T
 2 Call               CEEPLPKA n/a      CEECRINI E+B0A                   CEE.SC
 3 Call               CEEPLPKA n/a      CEECRINV E+302                   CEE.SC
 4 Link               RDBKC01  RDBKC01  RDBKC01  L#358 P+A9E E+A9E       REDBOO
 5 Call               DFHAPLI  DFHAPLI1 n/a      P+2E6E                  CICS.T
 6 Call               CEEPLPKA n/a      CEECRINI E+B0A                   CEE.SC
 7 Call               CEEPLPKA n/a      CEECRINV E+480                   CEE.SC
 8 Call               CEEEV011 n/a      CEEEV011 E+204                   CEE.SC
 9 Call               CEEEV011 n/a      IBMPMINV E+4DE                   CEE.SC
10 EXEC CICS  ***** RDBKP01  RDBKP01  RDBKP01  F#0 L#22 P+2AC E+24C      REDBOO
11 Abend AEXL        DFHAIP   DFHEIP   n/a      P+24E6                  CICS.T


(*) One or more of the following abbreviations might appear in the "Event
```

Figure 5-5 Event summary

The Event Summary display shows us that the RDBKC01 COBOL program in event 4 issued an EXEC CICS LINK at line 358 to the RDBKP01 PL/I program in event 10, which abnormally terminated online 22 with an AEXL abend.

4. Select event 10 to show the point of failure in more detail, as shown in Figure 5-6.

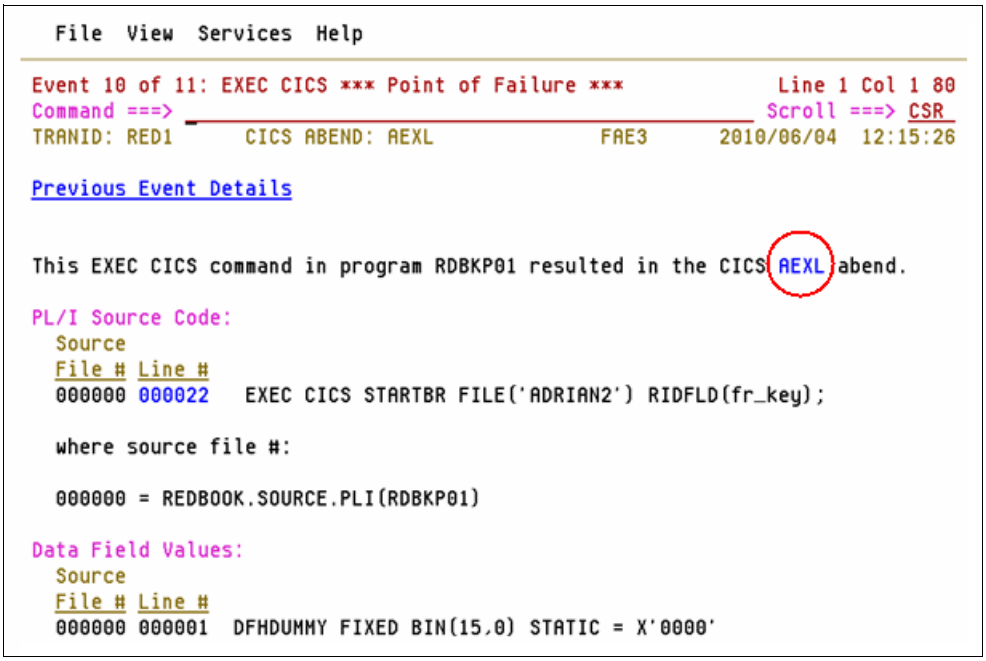


Figure 5-6 Point of failure

Here we can see that the abending program statement was an EXEC CICS STARTBR from file ADRIAN2.

5. Place the cursor on the **AEXL** point-and-shoot field (which is circled in red in Figure 5-6) and press Enter to show the meaning of the CICS AEXL abend, as shown in Figure 5-7.

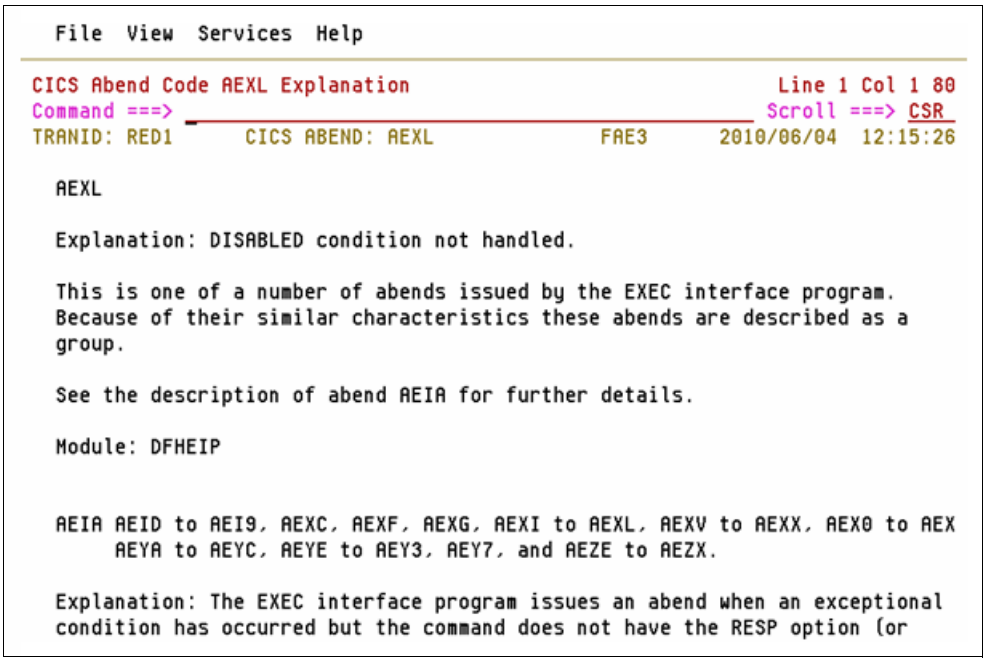


Figure 5-7 AEXL abend details

By scrolling through the abend code explanation, we can see that the AEXL abend is related to the processing of a disabled file.

We can now conclude that the problem that caused the AEXL abend was that the file ADRIAN2 was disabled. An example of how to fix this problem is shown in “Using CICS Explorer to fix the problem” on page 162.

5.1.2 Starting Fault Analyzer by using CICS Explorer

Complete the following steps to start Fault Analyzer by using CICS Explorer:

1. Start CICS Explorer and ensure that the Fault Analyzer perspective is selected. Select **Window** → **Open Perspective** → **Other**, then select **Fault Analyzer perspective** from the list, as shown in Figure 5-8.

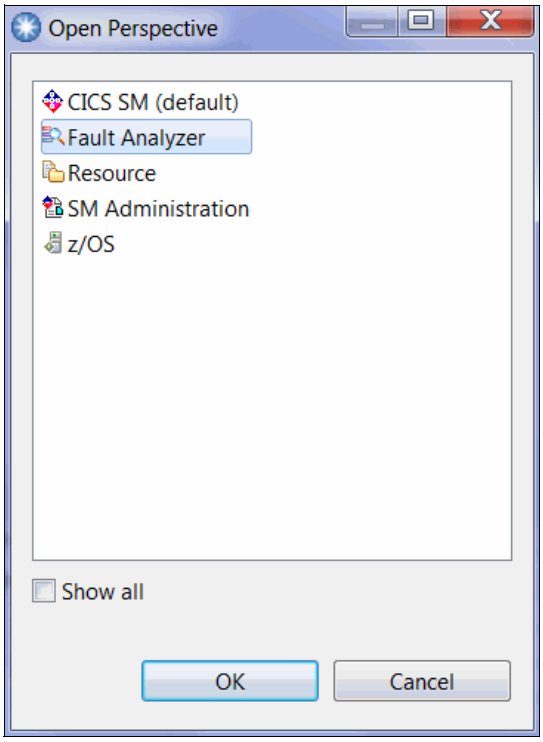


Figure 5-8 Select Fault Analyzer perspective

2. With the Fault Analyzer perspective selected, find the history file that was used for the CICS transaction abend in the Fault Analyzer objects view. If it is the first time this particular history file is being used, then first right-click the view and select **Add history file**, as shown in Figure 5-9.

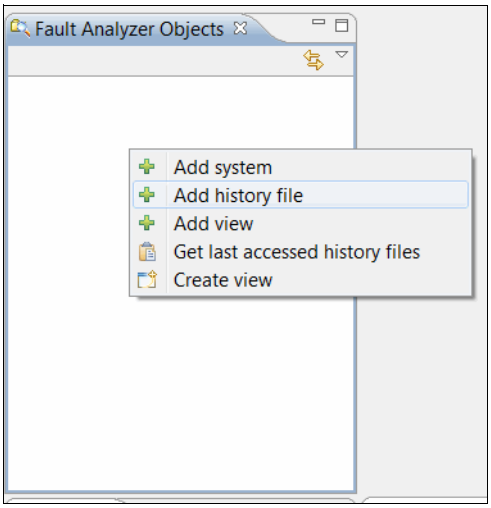


Figure 5-9 Add new history file

3. In the dialog (as shown in Figure 5-10), select a system name and specify the name of your history file or view.

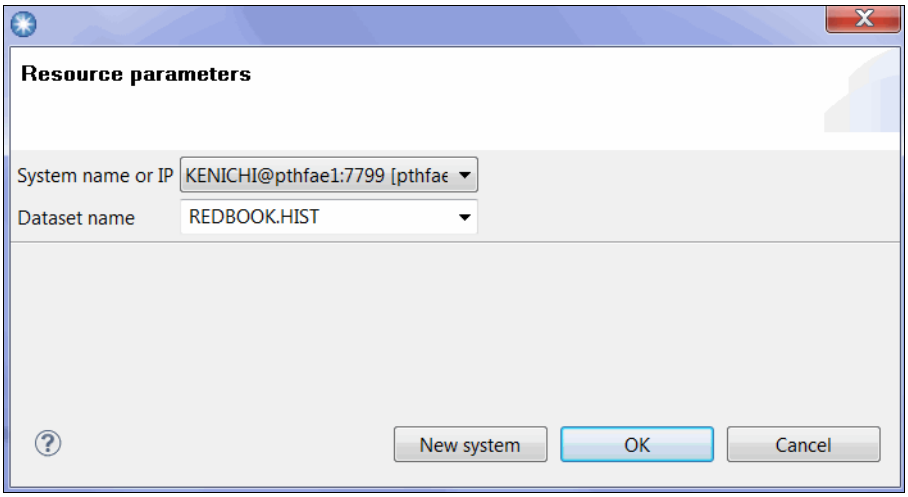


Figure 5-10 New history file dialog

4. With the history file details provided, click the history file name in the artifacts view, as shown in Figure 5-11.

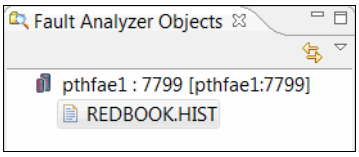


Figure 5-11 Fault Analyzer Objects view

5. A list of the history file fault entries is shown in the lower right view, as shown in Figure 5-12.

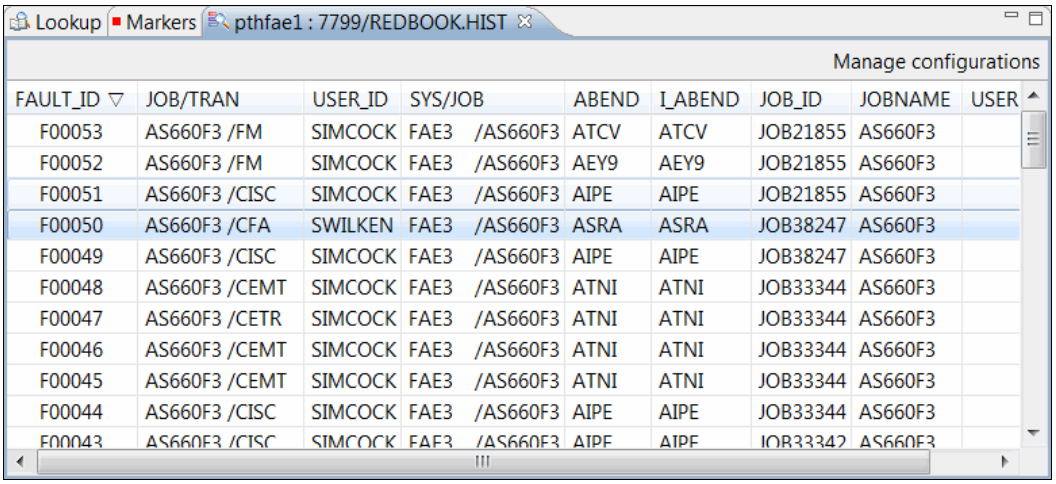


Figure 5-12 View showing the contents of a history file

6. To update the fault entry list, right-click the history file name and select **Refresh History File/View**.
7. Right-click the **F00013** fault entry and select **Open**, as shown in Figure 5-13. Alternatively, double-click the fault entry to display the report.

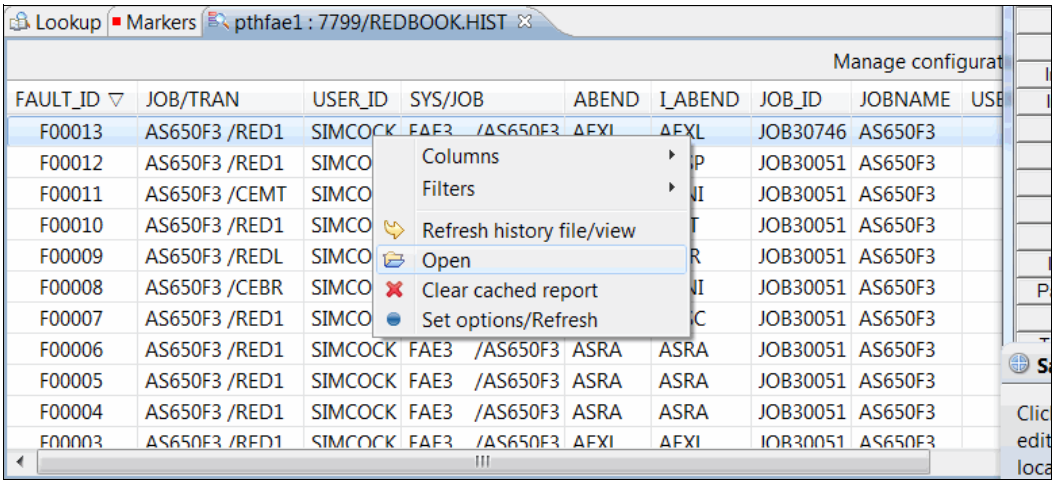


Figure 5-13 Open report

8. The report is displayed in the client, as shown in Figure 5-14.

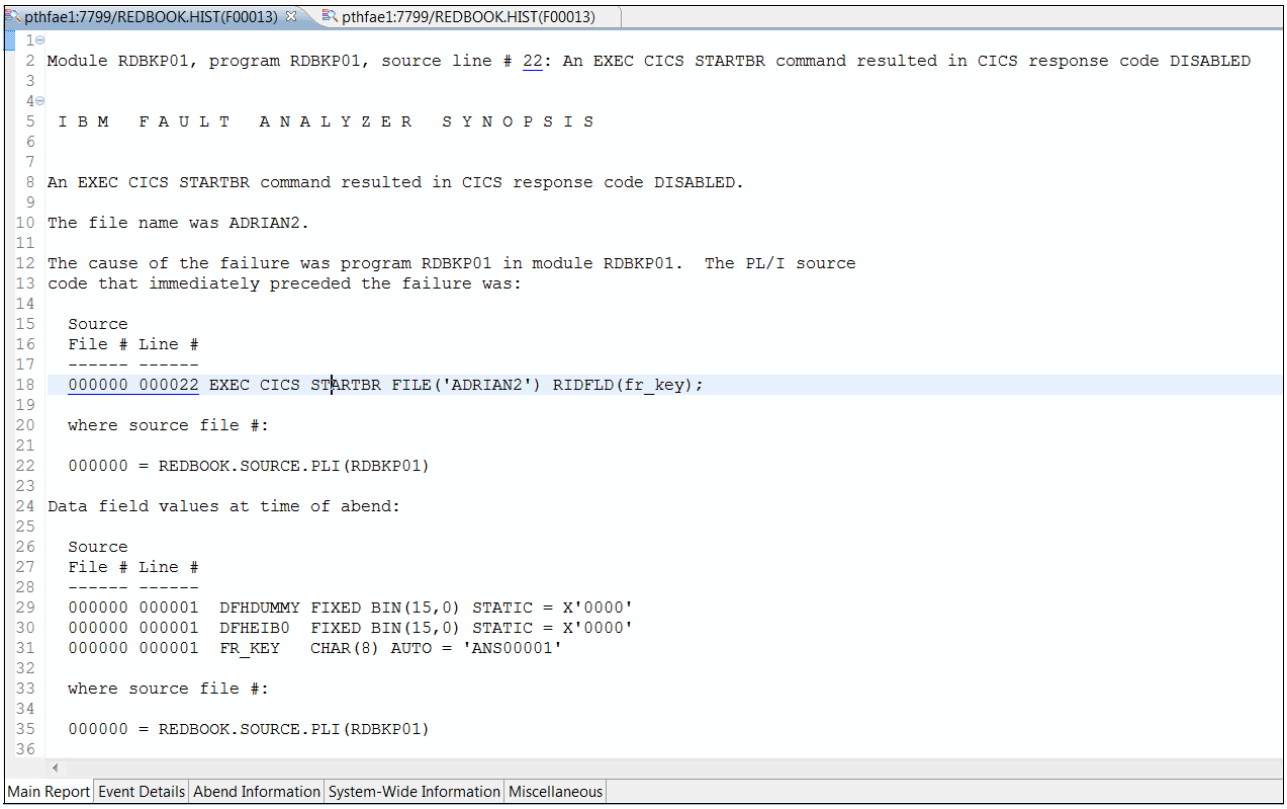


Figure 5-14 Fault Analyzer report summary

By selecting the various sections of the report from the tabs at the bottom of this view, we can see the same information about the abend that we saw when Fault Analyzer was used through the CICS interface.

5.1.3 Using CICS Explorer to fix the problem

Because we are already using CICS Explorer, it is convenient to also use this interface to fix the problem that caused the abend. Complete the following steps to fix the problem:

1. Select the **CICS SM** perspective and establish a connection to the appropriate CICS region.
2. Select the **Files** tab and locate file name ADRIAN2, which is shown as being disabled. Right-click this file and select **Enable**, as shown in Figure 5-15.

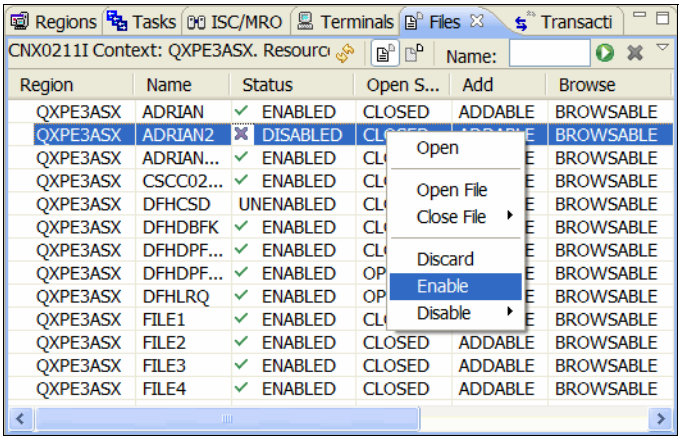


Figure 5-15 Enable file

The Perform ENABLE Operation window is displayed, as shown in Figure 5-16.

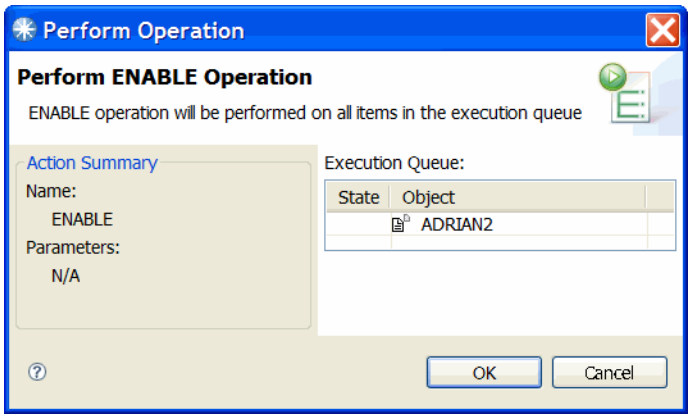


Figure 5-16 Perform enable option

3. Click **OK** to enable operation. The status of the ADRIAN2 file is changed to enabled, as shown in Figure 5-17.

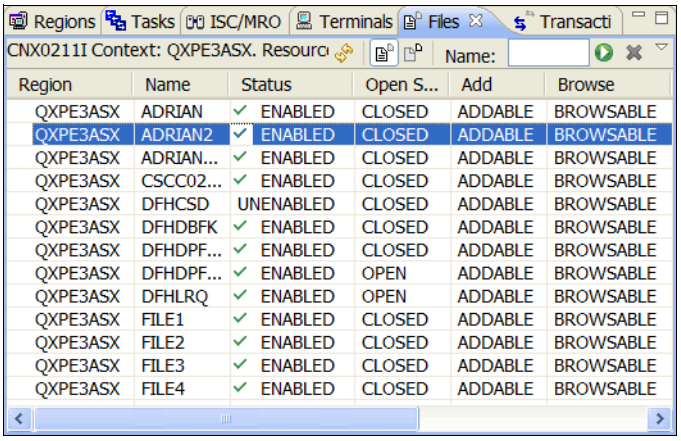


Figure 5-17 File is enabled

We can now rerun our abending transaction.

5.2 Using File Manager with CICS

In this section, we present examples of the use of File Manager with CICS.

5.2.1 Test messages

After the ADRIAN2 file is enabled, rerunning the RED1 CICS transaction `Test Number 1 - Open File error` test results in the message that is shown in Example 5-1 being displayed on the CICS terminal.

Example 5-1 Results

```
Data record with key ANS00001 has 0 int value. Press ENTER to continue...
```

To correct the data record, we use File Manager. This process is described next.

5.2.2 Starting File Manager from a CICS terminal

Complete the following to start File Manager from the CICS terminal:

1. Enter the `FM` transaction name and press Enter.
2. Enter the user name when prompted.

   The File Manager for the CICS logon panel is displayed, as shown in Figure 5-18.

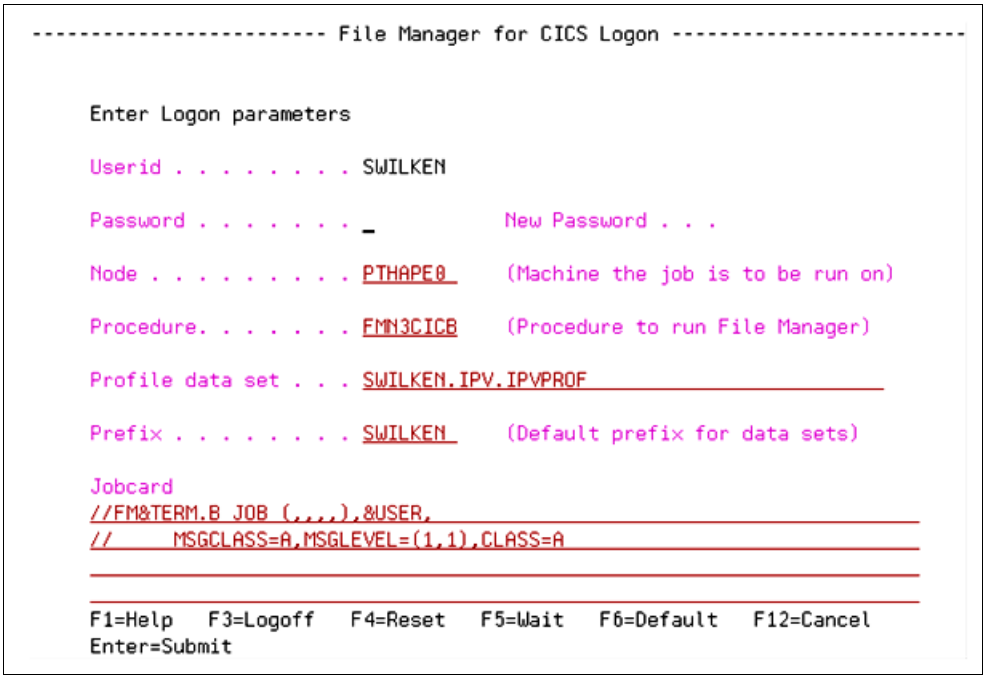


Figure 5-18 FM for CICS panel

3. Enter your password and press Enter.

The Primary Option menu is displayed, as shown in Figure 5-19.

```
 Process   Options   Help
FM/CICS                      Primary Option Menu

0  Settings      Set processing options              User ID . : SWILKEN
1  View          View data                           CICS User : SIMCOCK
2  Edit          Edit data                           CICS Appl : QXPE3ASX
3  Utilities     Perform utility functions           Date. . . : 2010/06/09
4  Templates     Template and copybook utilities     Time. . . : 10:21
5  WebSphere MQ  List, view and edit MQ data
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
FD FM/DB2        File Manager for DB2 z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 1  1. File
    2. Temporary Storage
    3. Transient Data

Command ===> 2
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 5-19 FM Primary option menu

4. Select option **2 Edit** and press Enter.

   The Edit CICS File Entry Panel is displayed, as shown in Figure 5-20.

```
 Process   Options   Help
FM/CICS                  Edit CICS File Entry Panel

Input CICS VSAM File:
   File name  . . . . adrian2
   Sysid  . . . . . ____
   Start position . . ________________________________ +
   Record limit . . . ________     Record sampling  _
   Inplace edit . . . _            Prevent inserts and deletes
   Lock resource  . . _            Name ____________________________________
Copybook or Template:
   Data set name  . . 'redbook.source.pli'
   Member . . . . . . rdbkp01___   (Blank or pattern for member list)
Processing Options:
 Copybook/template   Start position type   Enter "/" to select option
 1  1. Above         _  1. Key             _  Edit template  _ Type (1,2,S)
    2. Previous         2. RBA             _  Include only selected records
    3. None             3. Record number   _  Create audit trail
    4. Create dynamic

Command ===> ____________________________________________________________
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 5-20 CICS file entry panel

5. On the Edit CICS File Entry Panel, enter the following information:
   - File name: `ADRIAN2`
   - Data set name: `REDBOOK.SOURCE.PLI`
   - Member: `RDBKP01`

   Press Enter. The contents of the data set that are associated with the ADRIAN2 file name are displayed, as shown in Figure 5-21.

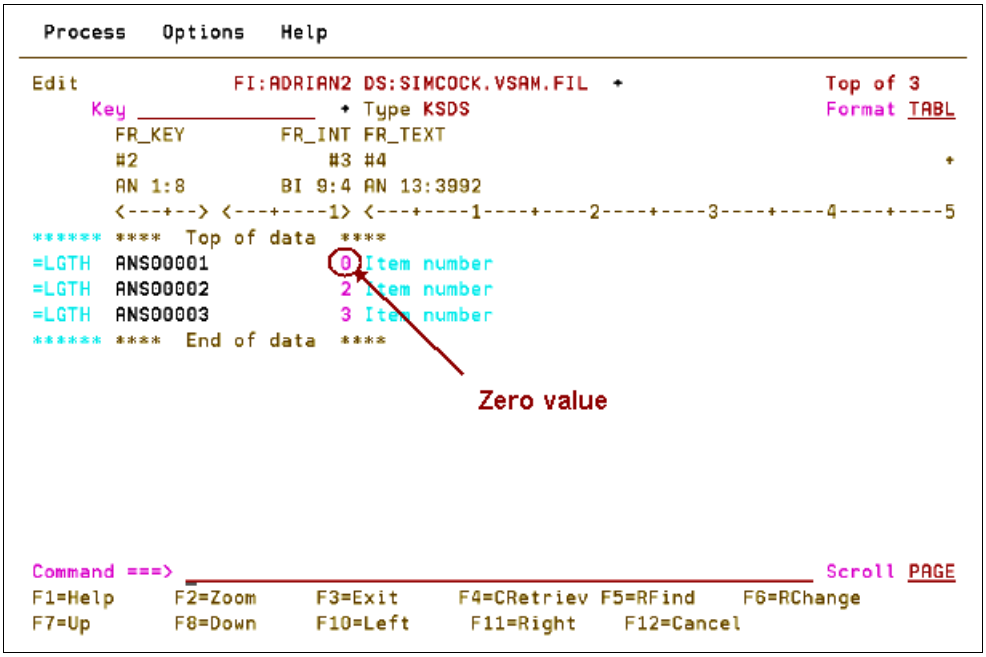


Figure 5-21 Contents of data set

As shown in Figure 5-21, the value in the first record is 0 (circled in red).

6. To correct this situation, we change the value to 1, as shown in Figure 5-22.

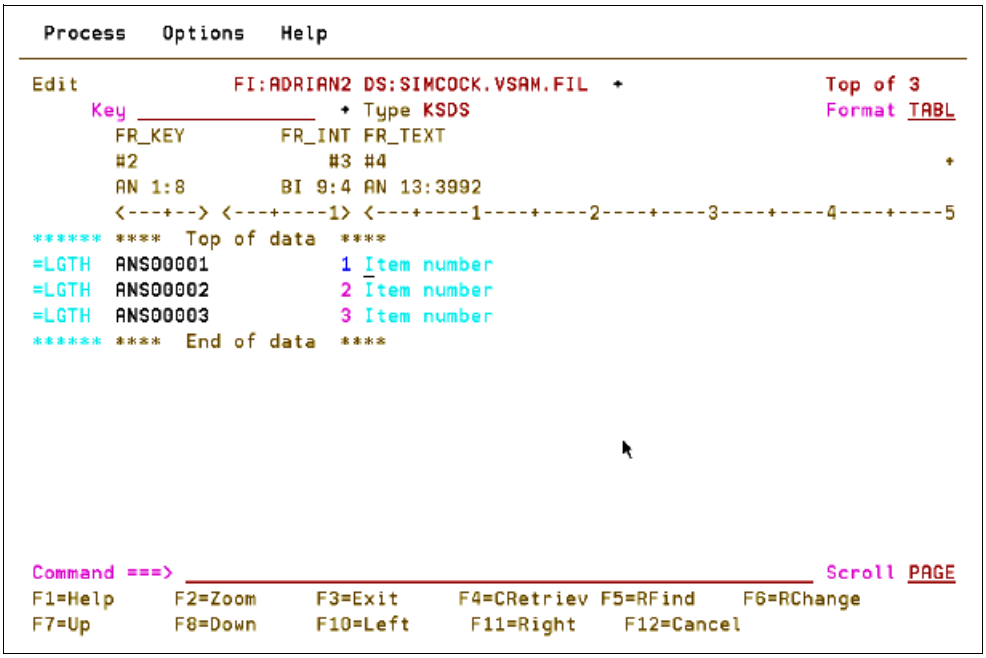


Figure 5-22 Change from 0 to 1

7. Press PF3 repeatedly to save the update and then exit File Manager for CICS.
8. Rerun the RED1 CICS transaction `Test Number 1 - Open File error` test.

   A success message is displayed, as shown in Example 5-2.

Example 5-2 Results

```
All data correctly validated..
```

5.2.3 Accessing CICS resources from ISPF

In addition to the steps described in the previous section, you can access CICS resources directly from the File Manager Base product under ISPF. As you can see in Figure 5-23, you can selection option 10 to access the panels to specify CICS resources.

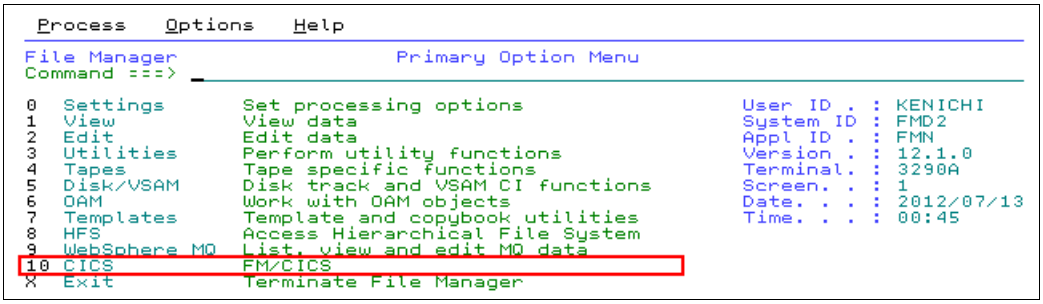


Figure 5-23 File Manager primary option menu with CICS option

If you select option 10, you are in the FM/CICS Primary Option menu, which is shown in Figure 5-24.

```
 Process    Options    Help
FM/CICS                          Primary Option Menu
Command ===> _

1  View           View data                                  User ID . : KENICHI
2  Edit           Edit data                                  Date. . . : 2012/07/13
3  Print          Print data                                 Time. . . : 00:50
4  List           List resources
X  Exit           Terminate FM/CICS

Processing Options:
 CICS Resource
 1   1. File
     2. Temporary Storage
     3. Transient Data
```

Figure 5-24 FM/CICS Primary Option menu that is invoked from the File Manager Primary Option panel

If you select option 1, you see the view options panel, which is shown in Figure 5-25. The panel displays the currently selected CICS application ID at the upper right corner. You can specify a different CICS application ID, or specify a pattern to look up an ID. If you specify a pattern (for example, enter * to look up every application ID that is known to the File Manager on your system), it brings up the application ID lookup panel, as shown in Figure 5-26.

```
 Process    Options    Help
FM/CICS                    View CICS File Entry Panel
Command ===> _                                              Applid C62D2FM5

Input CICS VSAM File:
   File name  . . . . ________
   Sysid  . . . . . . ____
   Start position . . ________________________________ +
   Record limit . . . ________      Record sampling  _
Copybook or Template:
   Data set name  . . ____________________________________________
   Member . . . . . . __________    (Blank or pattern for member list)
Processing Options:
 Copybook/template   Start position type   Enter "/" to select option
 3   1. Above          _  1. Key             _  Edit template   _  Type (1,2,S)
     2. Previous       _  2. RBA             _  Include only selected records
     3. None              3. Record number
     4. Create dynamic    4. Formatted key
```

Figure 5-25 FM/CICS view options panel

```
 Process    Options    Help
 File Manager          CICS Applid Selection list          Row 00001 of 00024
Command ===> _                                                Scroll CSR

  Applid   Status   Description
  *        *        *
_ C62D2FA1 Inactive V7  Prod level V7, Fault analyser
_ C62D2FM1 Active   SCLM testing system 1
_ C62D2FM2 Inactive SCLM testing system 2 - V11 TYRONED
_ C62D2FM3 Inactive V9  Latest driver
_ C62D2FM4 Inactive V8  Latest driver
_ C62D2FM5 Active   V11 CICS Gui support driver PM47010
_ C63D2FM1 Active   V11 AA47010
_ C63D2FM2 Inactive V11 AA47010
_ C63D2FM3 Inactive V11 AA47010
_ C63D2FM4 Inactive V11 AA47010
_ C63D2FM5 Inactive V11 AA47010
_ C64D2FA1 Inactive V7  Prod level V7, Fault Analyser
_ C64D2FM1 Inactive V12 Service Drivers
_ C64D2FM2 Inactive V10 SCLM testing currently MIKEMOR1
_ C64D2FM3 Inactive V10 Latest driver CICS
_ C65D2FM1 Active   V12 Latest driver CICS
_ C65D2FM2 Active   V12 Latest driver CICS
_ C65D2FM3 Inactive V10 SCLM - CARLAND - FMN10SVC.LISTENER.AUTH1
_ C65D2FM4 Inactive SCLM testing system 4 - V12 SOPERW
_ C65D2FM5 Inactive SCLM testing system 5 - V12 Carland
_ C66D2FM1 Active   V11 Latest driver CICS
_ C66D2FM4 Inactive V11 SCLM - CARLAN1
_ C67D2FM1 Inactive V11 CICSTS 4.2 with V11 driver on FMD2
_ C67D3FM1 Inactive V11 CICSTS 4.2 with V11 driver on TSD3
****  End of data  ****
```

Figure 5-26 CICS application ID lookup and selection panel

The approach was used thus far is a panel-driven approach to identify a CICS resource and manipulate the selected resource. You also can specify a CICS resource in the FM/Base product by using the following syntax:

- `fi:applid:file_name`

  This syntax is used to specify a CICS file.

- `td:applid:td_queue_name`

  This syntax is used to specify a CICS Transient Data Queue.

- `ts:applid:ts_queue_name`

  This syntax is used to specify a CICS Temporary Storage Queue.

The FM/Base view options panel is shown in Figure 5-27. A CICS file name is specified for the view target resource.

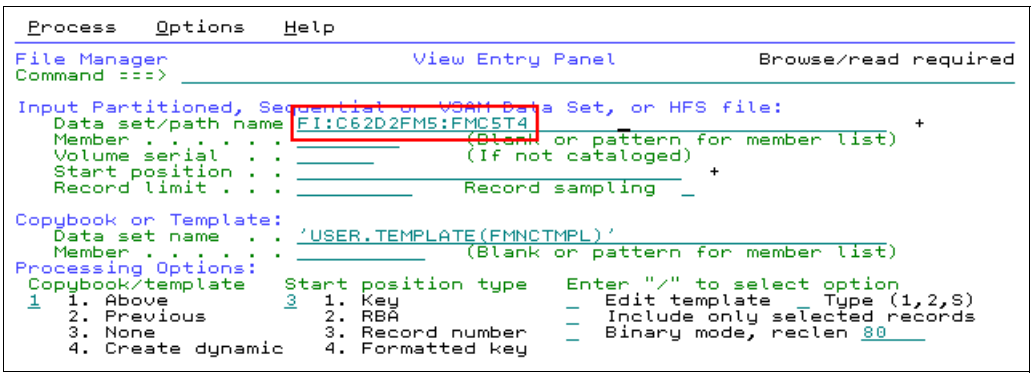


Figure 5-27 FM/Base view options panel with a CICS file as a target resource to view

5.3 Using Debug Tool with CICS

The test program that we use in this scenario consists of a COBOL program (RDBKC01), which calls a PL/I program (RDBKP02) that abends. The test is run from the RED1 transaction menu as shown in Figure 5-1 on page 153, by selecting Test Number 2 - Bad Data.

Figure 5-28 shows the CICS transaction that abnormally terminated with abend code ASRA.

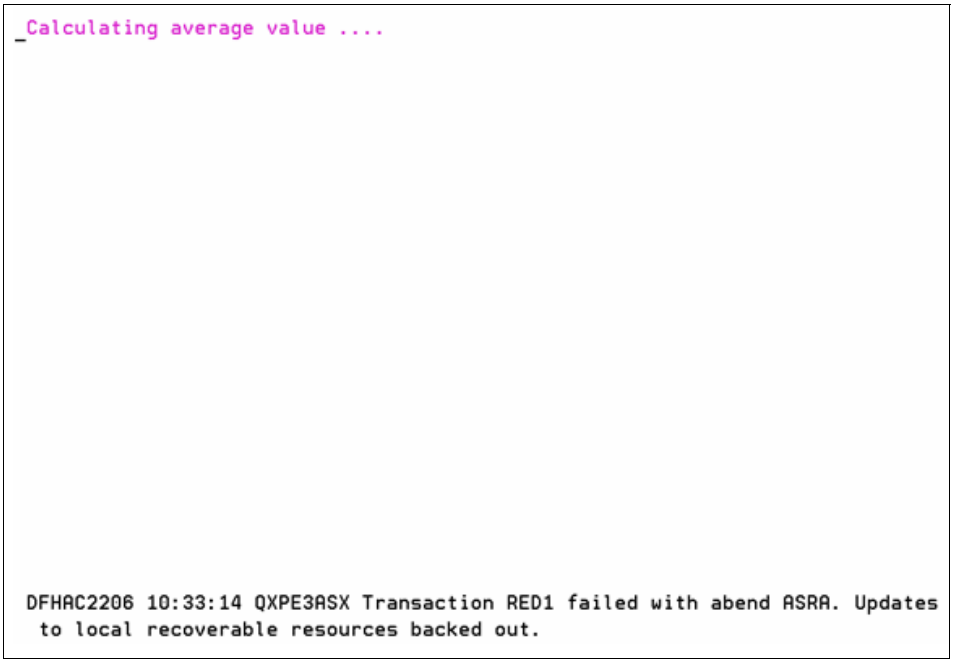


Figure 5-28 ASRA abend in CICS

By using Fault Analyzer (similar to the example shown in 5.1, “Using Fault Analyzer with CICS”), we determined that the problem is that data field `I` contained zero in a divide statement, as shown in Figure 5-29.

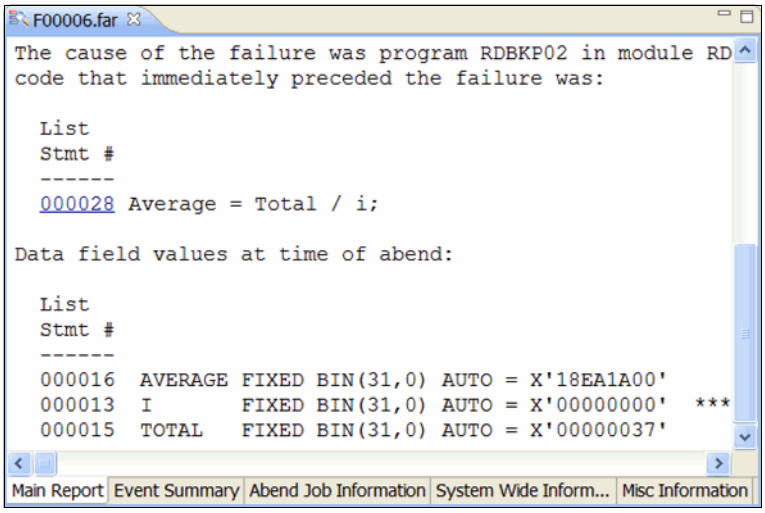


Figure 5-29 Fault in CICS

We use Debug Tool to determine how the data field `I` came to contain the zero value, which was started from the CICS terminal and CICS Explorer.

5.3.1 Starting Debug Tool from a CICS terminal

Before we can start Debug Tool to debug our abending transaction, we must set up a debugging profile. Complete the following steps to set up the profile:

1. To clear the window, enter the `DTCN` transaction name and press Enter.

   The Debug Tool CICS Control Primary menu is displayed, as shown in Figure 5-30.

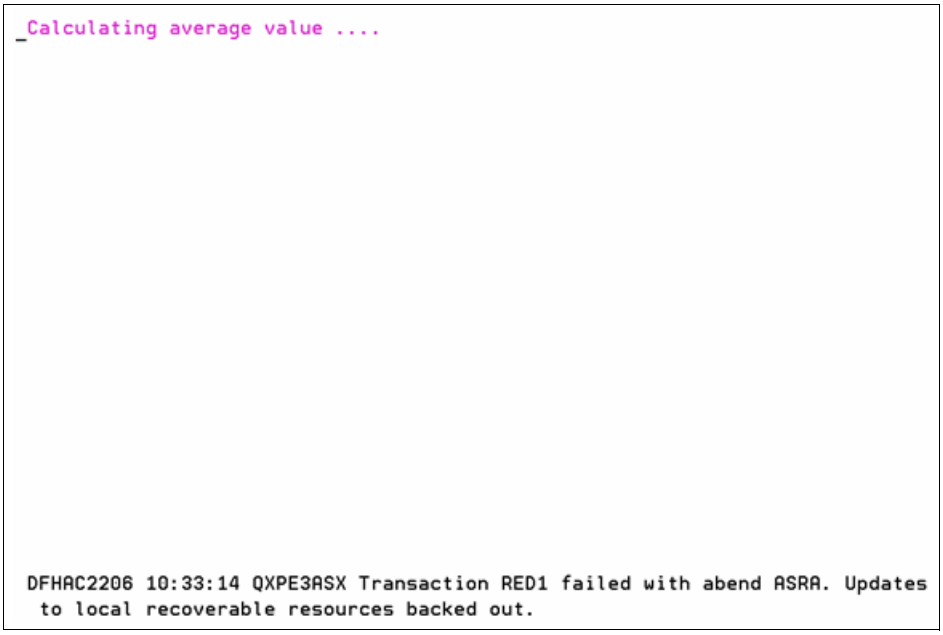


Figure 5-30 Debug Tool CICS Control Primary menu

2. In the Debug Tool CICS Control primary menu, enter the following information:
   - Transaction ID: `RED1`
   - LoadMod: `RDBKP02`
   - CU: *

   Terminal ID, user ID, and Display ID might be correct. If Terminal or Display ID is incorrect, place the cursor in the respective input field and press PF10.
3. Ensure that the specified Session Type is MFI, then press PF4 to save the profile. Press PF3 to exit.

   We are now ready to run the failing transaction.
4. Clear the window and enter the transaction ID.

Debug Tool is displayed, as shown in Figure 5-31.

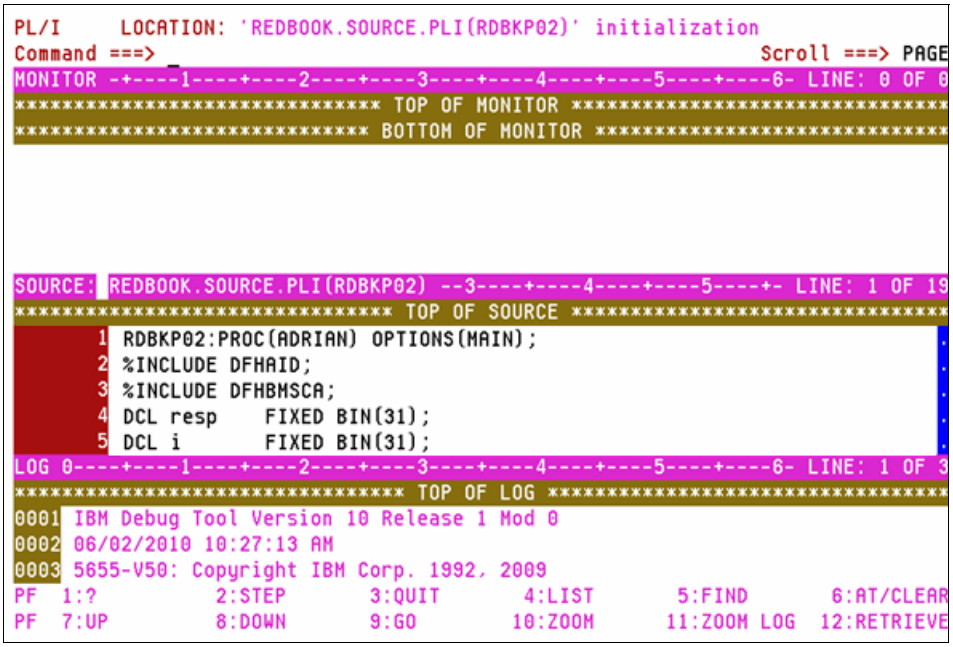


Figure 5-31 Debug Tool

Before we can set a breakpoint on the **I** data field, we must start the program.

5. Press PF2 to step forward once and then enter the command **AT CHANGE I WHEN I = 0**, as shown in Figure 5-32.

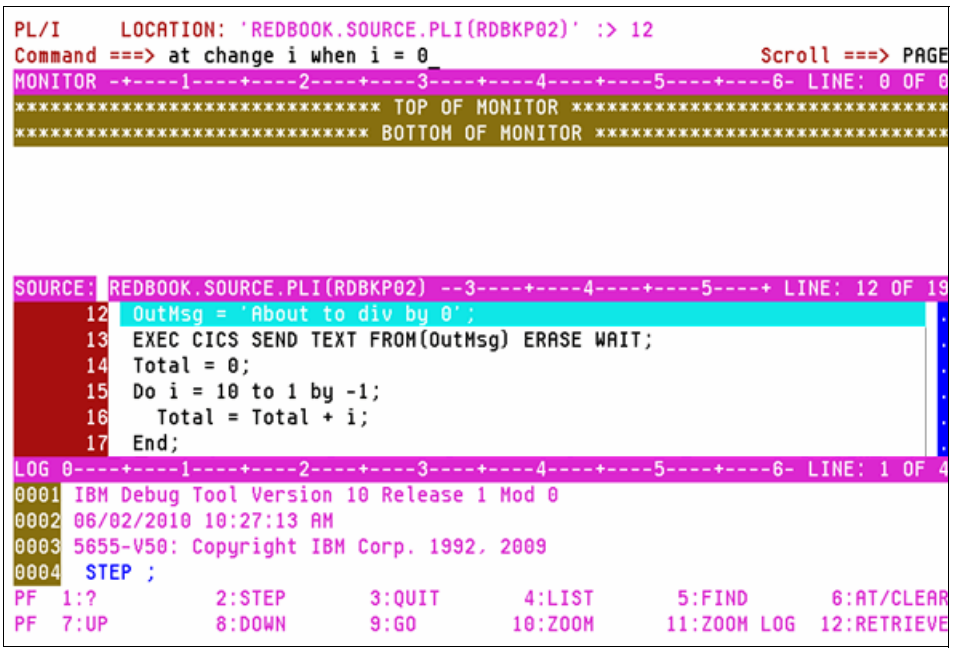


Figure 5-32 Command line enter, AT CHANGE I WHEN I = 0

6. Press PF9 to run the program until the value of the **I** data field reaches zero, which occurred on source line 18, as shown in Figure 5-33.

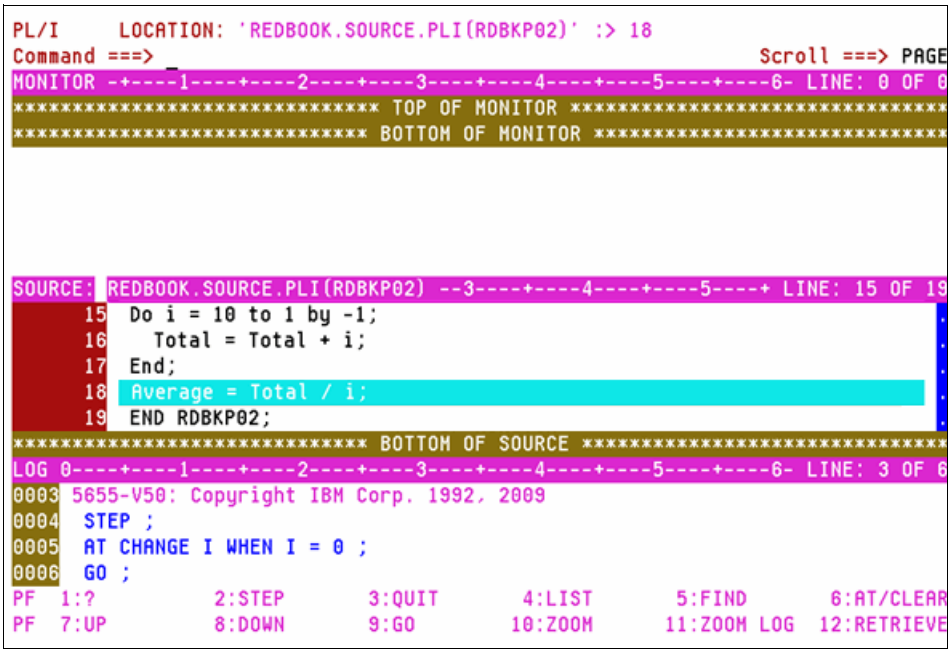


Figure 5-33 Condition matched

Notice that before this source line, there is a `DO` loop, which caused the **I** data field to be set to zero.

5.3.2 Starting Debug Tool by using CICS Explorer

Before we can use CICS Explorer to start Debug Tool for our abending transaction, we must deactivate and then reactivate the CICS debugging profile. Use the following steps to complete this process:

1. As in 5.3.3, “Starting Debug Tool from a CICS terminal”, we start the DTCN transaction, which produces the display that is shown in Figure 5-34.

```
DTCN              Debug Tool CICS Control - Primary Menu              QXPE3ASX
                           * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id      ==> 1260
 Transaction Id   ==> red1
 LoadMod::>CU(s)  ==> rdbkp02  ::> *             ==>          ::>
                  ==>          ::>               ==>          ::>
                  ==>          ::>               ==>          ::>
                  ==>          ::>               ==>          ::>
 User Id          ==> swilken
 NetName          ==>
 IP Name/Address  ==>
Select type and ID of debug display device
 Session Type     ==> tcp                 MFI, TCP
 Port Number      ==> 8899                TCP Port
 Display Id       ==> 192.168.129.231_

Generated String:  TEST(ALL,'*',PROMPT,'TCPIP&192.168.129.231%8899:*')

Repository String: TEST(ALL,'*',PROMPT,'TCPIP&192.168.129.231%8899:*')

Profile Status:    Active. Press PF5 to Inactivate.

PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
```

Figure 5-34 DTCN transaction started

2. Specify the following information:
   - Session Type: `TCP`
   - Port Number: Port number greater than `5000`
   - Display ID: The correct IP name or address for the CICS Explorer workstation
3. Press PF4 to save and press PF3 to exit.
4. Start CICS Explorer and select the Debug perspective as was done when the Fault Analyzer perspective was selected in 5.1.2, “Starting Fault Analyzer by using CICS Explorer” on page 158.
5. In the Debug view, click the **Debug** UI daemon listener icon and select **Change Port**, as shown in Figure 5-35.

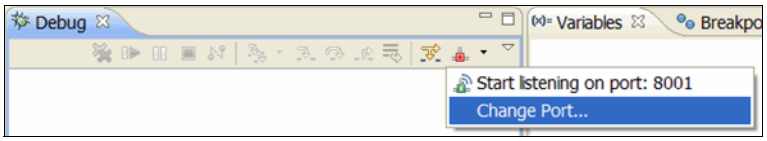


Figure 5-35 Change Port

The Preferences window is displayed, as shown in Figure 5-36.

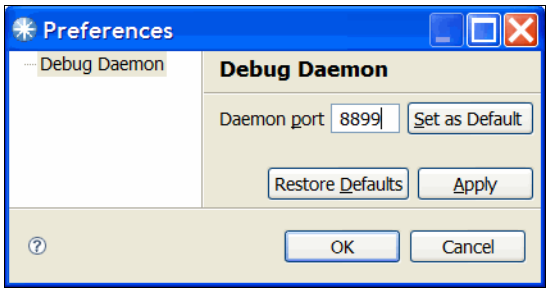


Figure 5-36 Preferences

6. Enter the port number for the CICS region (in this case, 8899) and click **OK**. Click the **Debug** UI daemon listener icon again and select the option to listen on this port number, as shown in Figure 5-37.

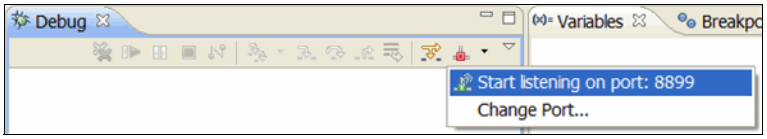


Figure 5-37 New Port 8989

7. Start the abending transaction from the CICS terminal.

The CICS Explorer Debug Tool perspective initializes and looks similar to the window that is shown in Figure 5-38.

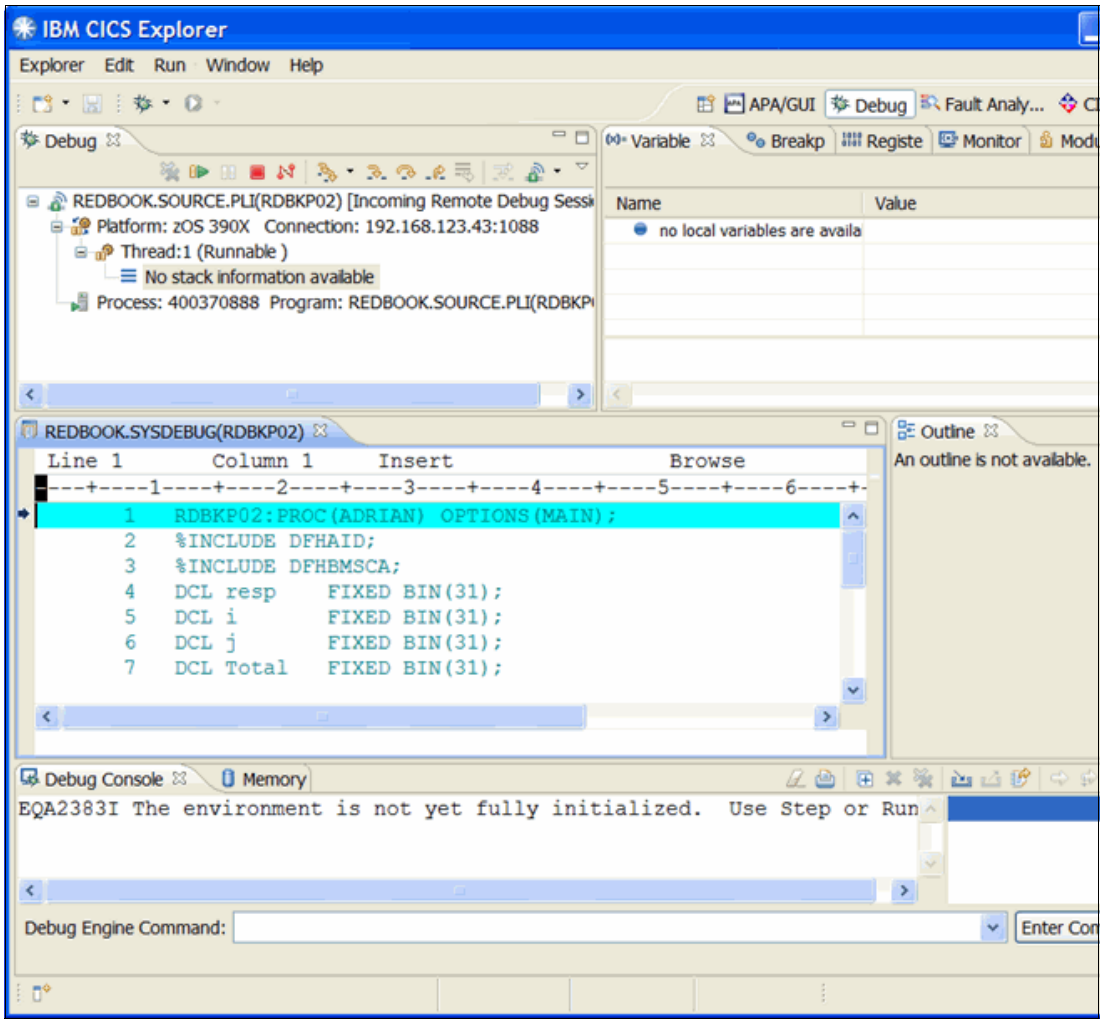


Figure 5-38 CICS Explorer Debug Tool perspective started

8. Press PF5 to step into the program and click the **Breakpoints** tab in the top-right view.

9. Select **Add Breakpoint** and then **Watch,** as shown in Figure 5-39.

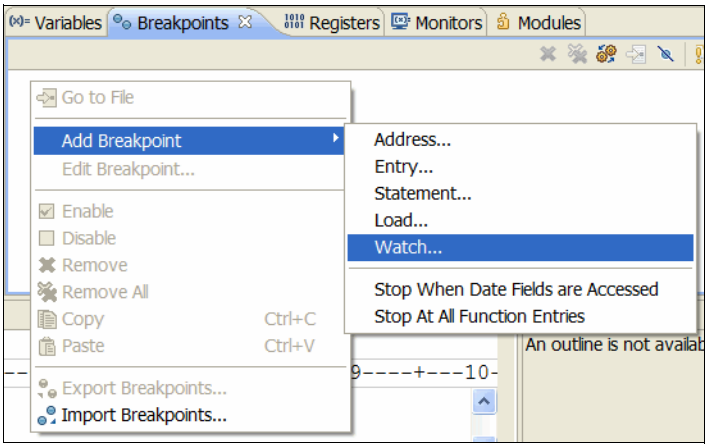


Figure 5-39 Add Watch breakpoint

The Add a Watch Breakpoint window is displayed, as shown in Figure 5-40.

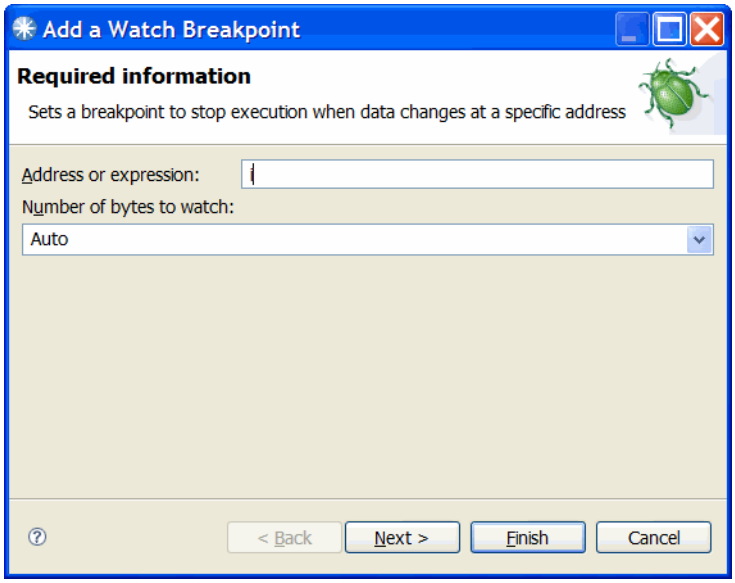


Figure 5-40 Add a watch breakpoint

10. Enter `I` in the Address or expression field. Click **Next**.

The Add a Watch Breakpoint panel is displayed, in which you can enter optional parameters, as shown in Figure 5-41.

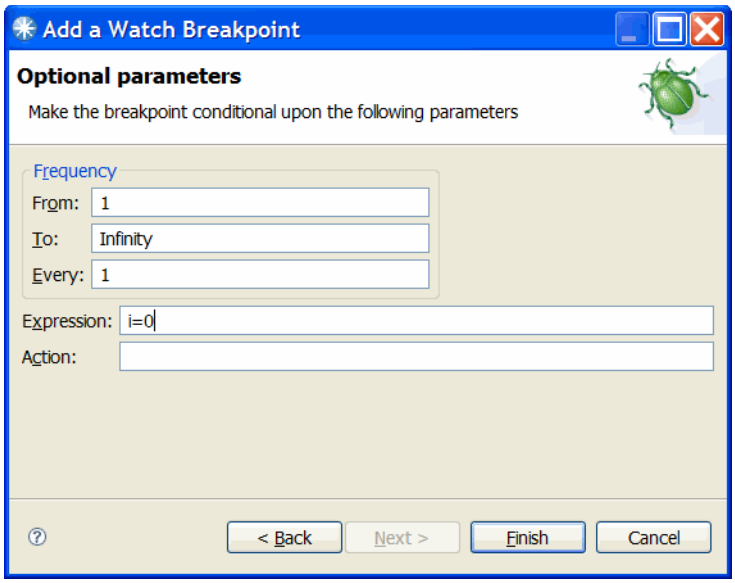


Figure 5-41 Enter i=0 in expression field

11. Enter **`I=0`** in the Expression field and click **Finish**.

    We have now set a breakpoint to stop running the program when the value of the data field **`I`** reaches zero.

12. Press PF8 to run the program.

    In a similar run when the CICS terminal interface to Debug Tool is used, the program stopped on line 18, as shown in Figure 5-42.

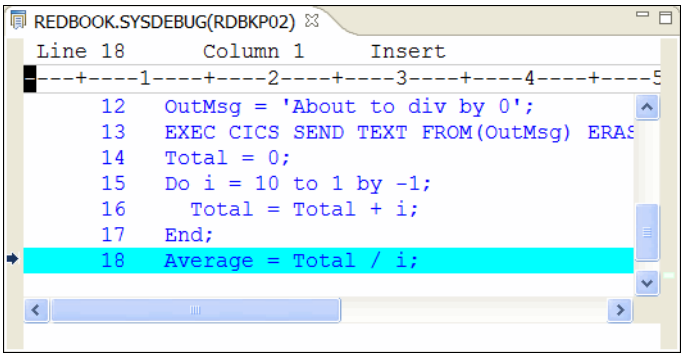


Figure 5-42 Stops on line 18

The test program that we use in this scenario consists of a COBOL program (RDBKC01), which calls a PL/I program (RDBKP02) that abends. The test is run from the RED1 transaction menu that is shown in Figure 5-1 on page 153, by selecting Test Number 2 - Bad Data.

Figure 5-43 shows the CICS transaction abnormally terminated with abend code ASRA.

```
_Calculating average value ....




DFHAC2206 10:33:14 QXPE3ASX Transaction RED1 failed with abend ASRA. Updates
 to local recoverable resources backed out.
```

Figure 5-43 ASRA abend in CICS

By using Fault Analyzer, similar to the example shown in 5.1, “Using Fault Analyzer with CICS”, we determined that the problem is that data field **I** contained zero in a divide statement, as shown in Figure 5-44.

```
15 The cause of the failure was program RDBKP02 in module RDBKP02.  The PL/I source
16 code that immediately preceded the failure was:
17
18   Source
19   File # Line #
20   ------ ------
21   000000 000018 Average = Total / i;
22
23   where source file #:
24
25   000000 = REDBOOK.SOURCE.PLI(RDBKP02)
26
27 Data field values at time of abend:
28
29   Source
30   File # Line #
31   ------ ------
32   000000 000001  AVERAGE FIXED BIN(31,0) AUTO = X'17E35220'
33   000000 000001  I       FIXED BIN(31,0) AUTO = X'00000000'
34   000000 000001  TOTAL   FIXED BIN(31,0) AUTO = X'00000037'
35
36   where source file #:
37
38   000000 = REDBOOK.SOURCE.PLI(RDBKP02)
```

Figure 5-44 Fault in CICS

We use Debug Tool to determine how the data field **I** came to contain the zero value, which was started from the CICS terminal and CICS Explorer.

5.3.3 Starting Debug Tool from a CICS terminal

Before we can start Debug Tool to debug our abending transaction, we must set up a debugging profile. Complete the following steps to start Debug Tool from a CICS terminal:

1. To clear the window, enter the `DTCN` transaction name and press Enter.

   The Debug Tool CICS Control primary menu is displayed, as shown in Figure 5-45.

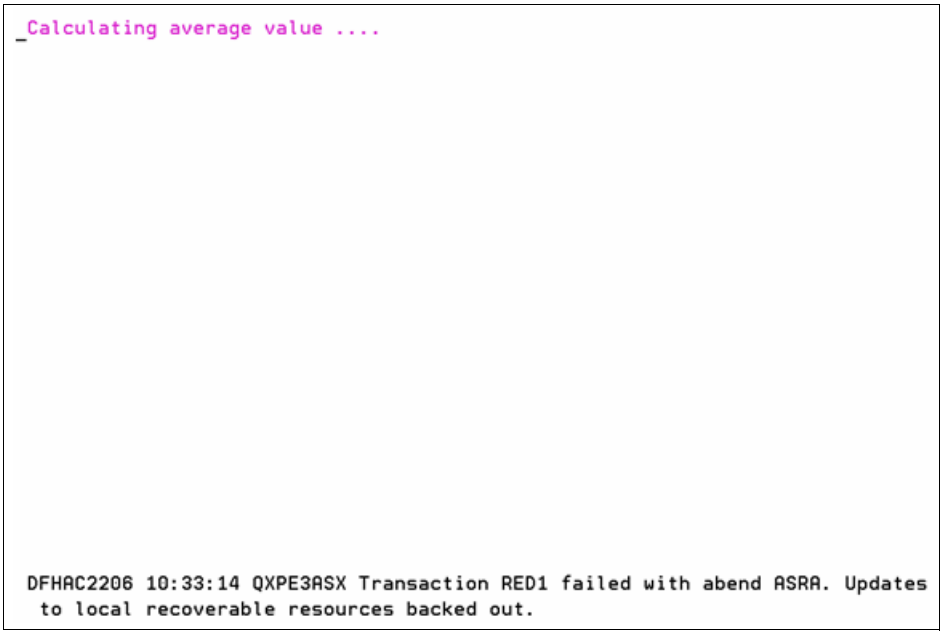


Figure 5-45 Debug Tool CICS Control Primary menu

2. In the Debug Tool CICS Control primary menu, enter the following information:
   - Transaction ID: `RED1`
   - LoadMod: `RDBKP02`
   - CU: *

   Terminal ID, User ID, and Display ID might be correct. If Terminal or Display ID is not correct, place the cursor in the respective input field and press PF10.

3. Ensure that the Session Type specified is MFI, then press PF4 to save the profile. Press PF3 to exit.

   We are now ready to run the failing transaction.

4. Clear the window and enter the transaction ID.

Debug Tool is displayed, as shown in Figure 5-46.

```
PL/I     LOCATION: 'REDBOOK.SOURCE.PLI(RDBKP02)' initialization
Command ===> _                                          Scroll ===> PAGE
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 0 OF 0
******************************** TOP OF MONITOR ********************************
******************************** BOTTOM OF MONITOR *****************************



SOURCE: REDBOOK.SOURCE.PLI(RDBKP02) --3----+----4----+----5----+- LINE: 1 OF 19
******************************** TOP OF SOURCE *********************************
      1 RDBKP02:PROC(ADRIAN) OPTIONS(MAIN);
      2 %INCLUDE DFHAID;
      3 %INCLUDE DFHBMSCA;
      4 DCL resp    FIXED BIN(31);
      5 DCL i       FIXED BIN(31);
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 3
******************************** TOP OF LOG ************************************
0001 IBM Debug Tool Version 10 Release 1 Mod 0
0002 06/02/2010 10:27:13 AM
0003 5655-V50: Copyright IBM Corp. 1992, 2009
PF  1:?         2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP        8:DOWN      9:GO        10:ZOOM     11:ZOOM LOG 12:RETRIEVE
```

Figure 5-46 Debug Tool

Before we can set a breakpoint on the **I** data field, we must start the program.

5. Press PF2 to step forward once and then enter the command **AT CHANGE I WHEN I = 0**, as shown in Figure 5-47.

```
PL/I     LOCATION: 'REDBOOK.SOURCE.PLI(RDBKP02)' :> 12
Command ===> at change i when i = 0_                    Scroll ===> PAGE
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 0 OF 0
******************************** TOP OF MONITOR ********************************
******************************** BOTTOM OF MONITOR *****************************



SOURCE: REDBOOK.SOURCE.PLI(RDBKP02) --3----+----4----+----5----+ LINE: 12 OF 19
     12  OutMsg = 'About to div by 0';
     13  EXEC CICS SEND TEXT FROM(OutMsg) ERASE WAIT;
     14  Total = 0;
     15  Do i = 10 to 1 by -1;
     16    Total = Total + i;
     17  End;
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 4
0001 IBM Debug Tool Version 10 Release 1 Mod 0
0002 06/02/2010 10:27:13 AM
0003 5655-V50: Copyright IBM Corp. 1992, 2009
0004  STEP ;
PF  1:?         2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP        8:DOWN      9:GO        10:ZOOM     11:ZOOM LOG 12:RETRIEVE
```

Figure 5-47 Command line enter, AT CHANGE I WHEN I = 0

6. Press PF9 to run the program until the value of the **I** data field reaches zero, which occurred on source line 18, as shown in Figure 5-48.

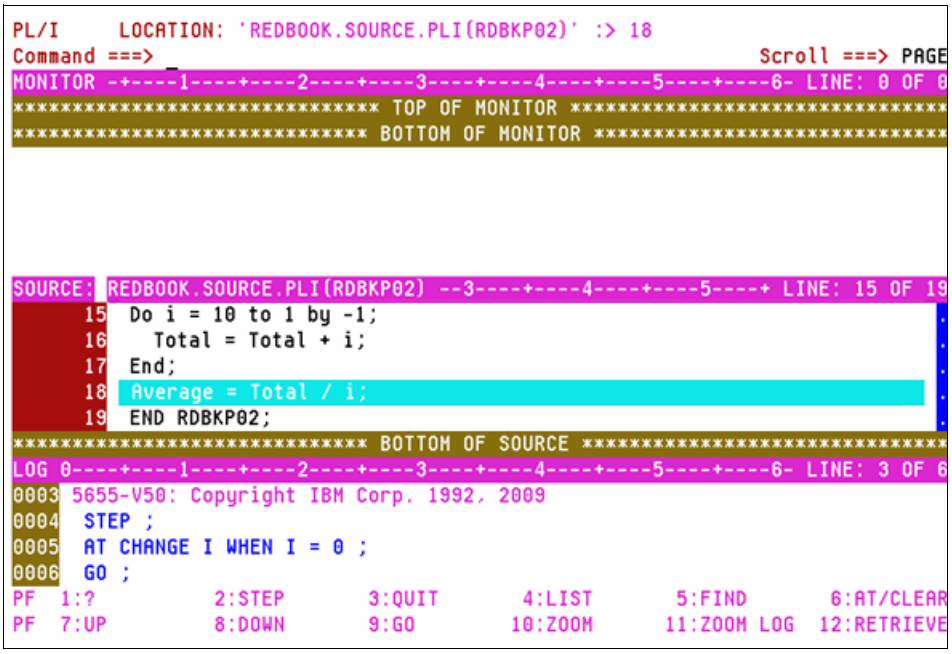


Figure 5-48 Condition matched

Before this source line, there is a `DO` loop, which caused the **I** data field to be set to zero.

5.3.4 Starting Debug Tool by using CICS Explorer

Before we can use CICS Explorer to start Debug Tool for our abending transaction, we must deactivate and then reactivate the CICS debugging profile. Use the following steps to complete this process:

1. As in 5.3.3, “Starting Debug Tool from a CICS terminal”, we start the DTCN transaction, which produces the display that is shown in Figure 5-49.

```
DTCN              Debug Tool CICS Control - Primary Menu              QXPE3ASX
                          * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id      ==> 1260
 Transaction Id   ==> red1
 LoadMod::>CU(s)  ==> rdbkp02  ::> *              ==>          ::>
                  ==>          ::>               ==>          ::>
                  ==>          ::>               ==>          ::>
                  ==>          ::>               ==>          ::>
 User Id          ==> swilken
 NetName          ==>
 IP Name/Address  ==>
Select type and ID of debug display device
 Session Type     ==> tcp                 MFI, TCP
 Port Number      ==> 8899                TCP Port
 Display Id       ==> 192.168.129.231_

Generated String:  TEST(ALL,'*',PROMPT,'TCPIP&192.168.129.231%8899:*')

Repository String: TEST(ALL,'*',PROMPT,'TCPIP&192.168.129.231%8899:*')

Profile Status:    Active. Press PF5 to Inactivate.

PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
```

Figure 5-49 DTCN transaction started

2. Enter the following information:
   - Session Type: `TCP`
   - Port Number: A port number greater than `5000`
   - Display ID: The correct IP name or address for the CICS Explorer workstation
3. Press PF4 to save and then press PF3 to exit.
4. Start CICS Explorer and select the Debug perspective, similar to when the Fault Analyzer perspective was selected in 5.1.2, “Starting Fault Analyzer by using CICS Explorer” on page 158.
5. In the Debug view, click the **Debug** UI daemon listener icon and select **Change Port**, as shown in Figure 5-50.

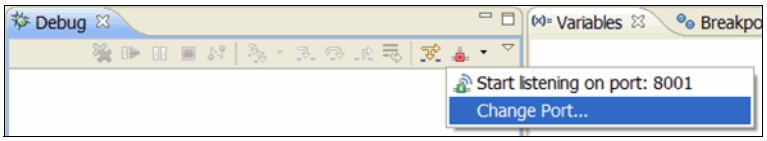


Figure 5-50 Change Port

The Preferences window is displayed, as shown in Figure 5-51.

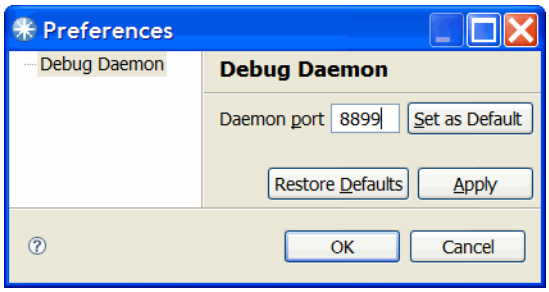


Figure 5-51 Preferences

6. Enter the port number for the CICS region (in our case, 8899) and click **OK**. Click the **Debug** UI daemon listener icon again and select the option to listen on this port number, as shown in Figure 5-52.

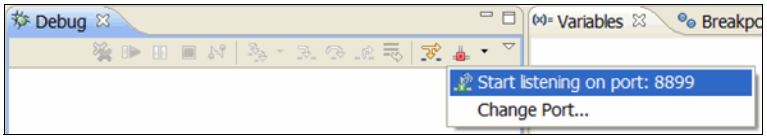


Figure 5-52 New Port 8989

7. Start the abending transaction from the CICS terminal.

The CICS Explorer Debug Tool perspective initializes and looks similar to the window that is shown in Figure 5-53.

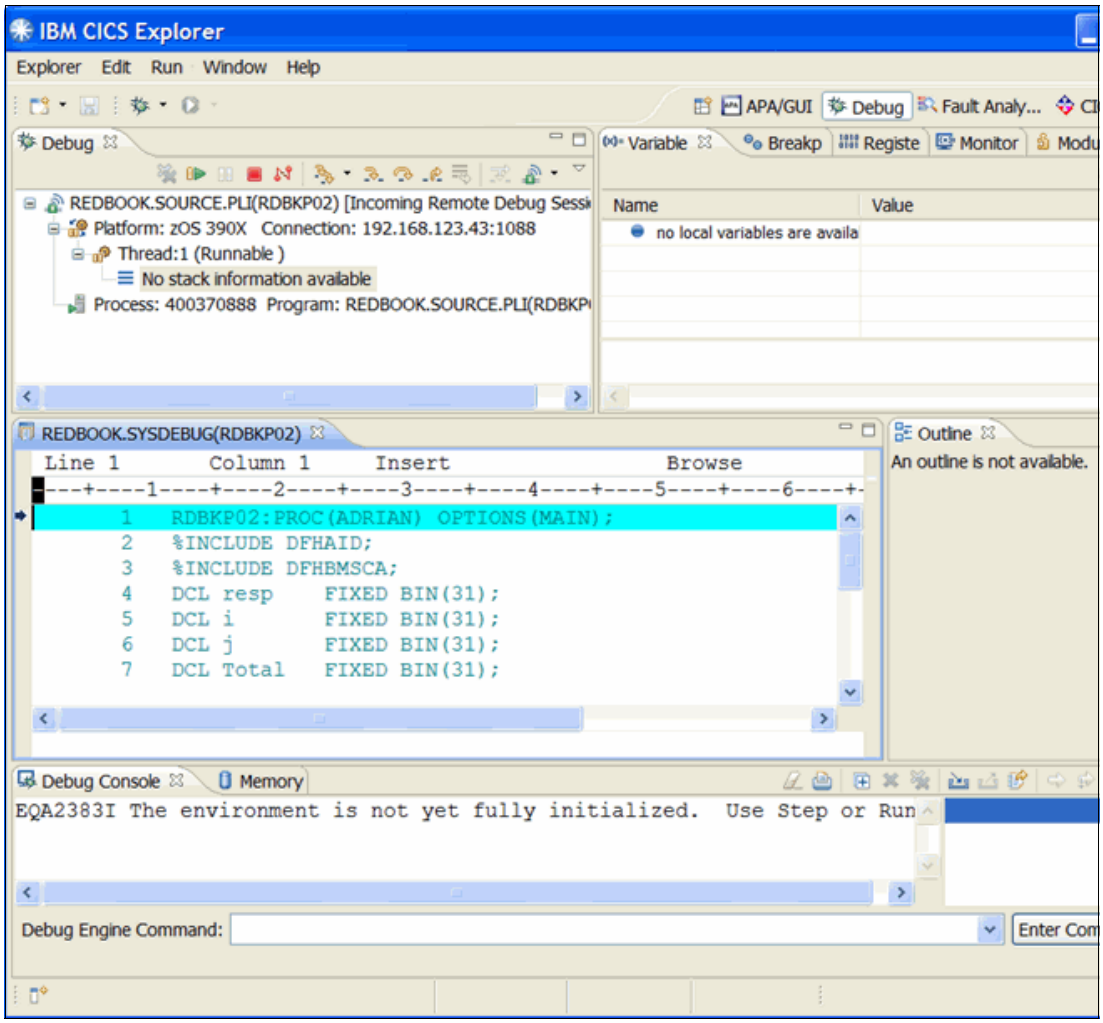


Figure 5-53 CICS Explorer Debug Tool perspective started

8. Press PF5 to step into the program and click the **Breakpoints** tab in the upper right view.

9. Select **Add Breakpoint** and then **Watch**, as shown in Figure 5-54.

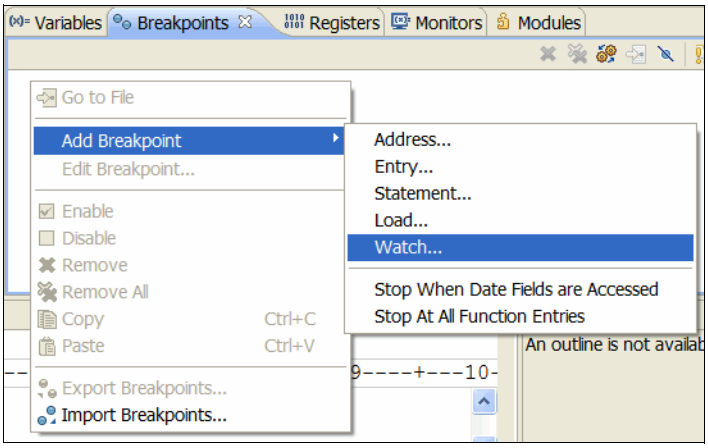


Figure 5-54 Add breakpoint

The Add a Watch Breakpoint window is displayed, as shown in Figure 5-55.

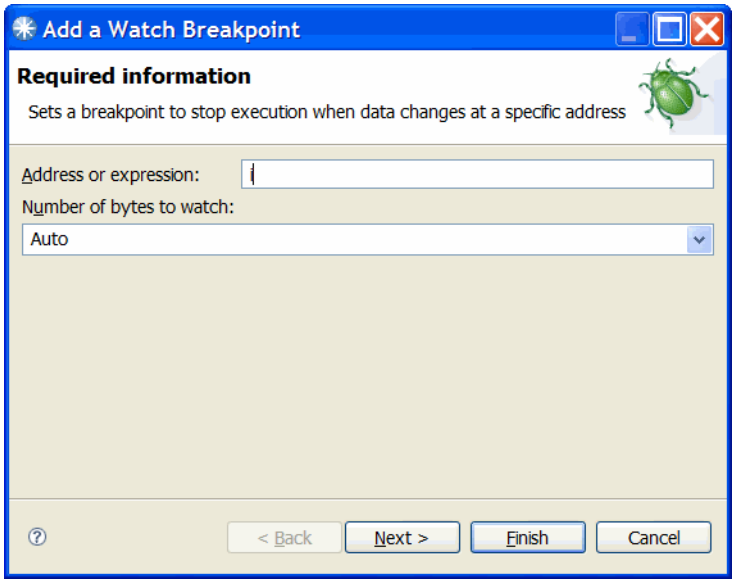


Figure 5-55 Add a watch breakpoint

10. Enter `I` in the Address or expression field and click **Next**.

The Add a Watch Breakpoint panel is displayed, in which you can enter optional parameters, as shown in Figure 5-56.

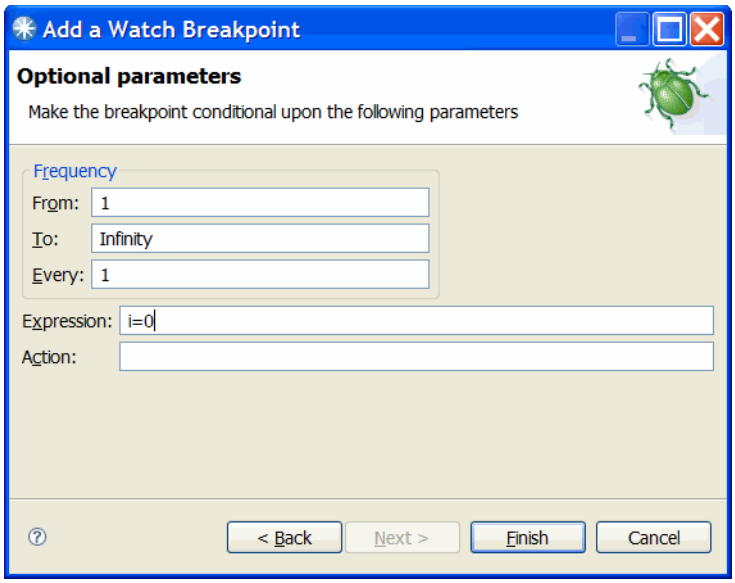


Figure 5-56 Enter i=0 in expression field

11. Enter **`I=0`** in the Expression field and click **Finish**.

    We have not set a breakpoint to stop the program when the value of the data field **`I`** reaches zero.

12. To run the program, press PF8.

    As when we used the CICS terminal interface to Debug Tool, the program stopped on line 18, as shown in Figure 5-57.

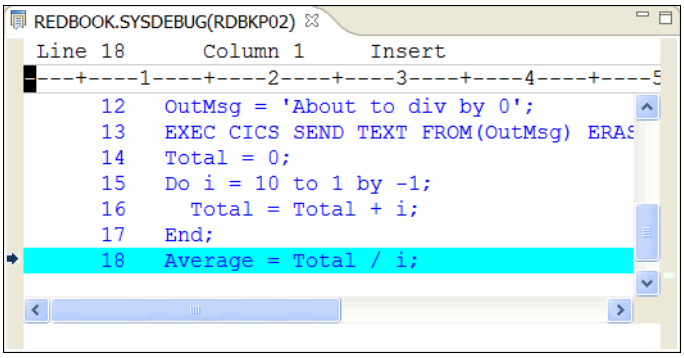


Figure 5-57 Stops on line 18

5.4 Using Application Performance Analyzer with CICS

The test program that we use in this scenario consists of a COBOL program (RDBKC01), which calls a PL/I program (RDBKP03) that loops. The test is run from the RED1 transaction menu that is shown in Figure 5-1 on page 153 by selecting Test Number 3 - Loop.

5.4.1 Starting Application Performance Analyzer from TSO/ISPF

For this scenario, before we start the looping CICS transaction, we must create an observation request. Complete the following steps to create this request:

1. Start APA from TSO/ISPF and enter `NEW` as a primary command, or typeover an existing request number. Press Enter.
2. Create an observation from the command line, as shown in Figure 5-58.

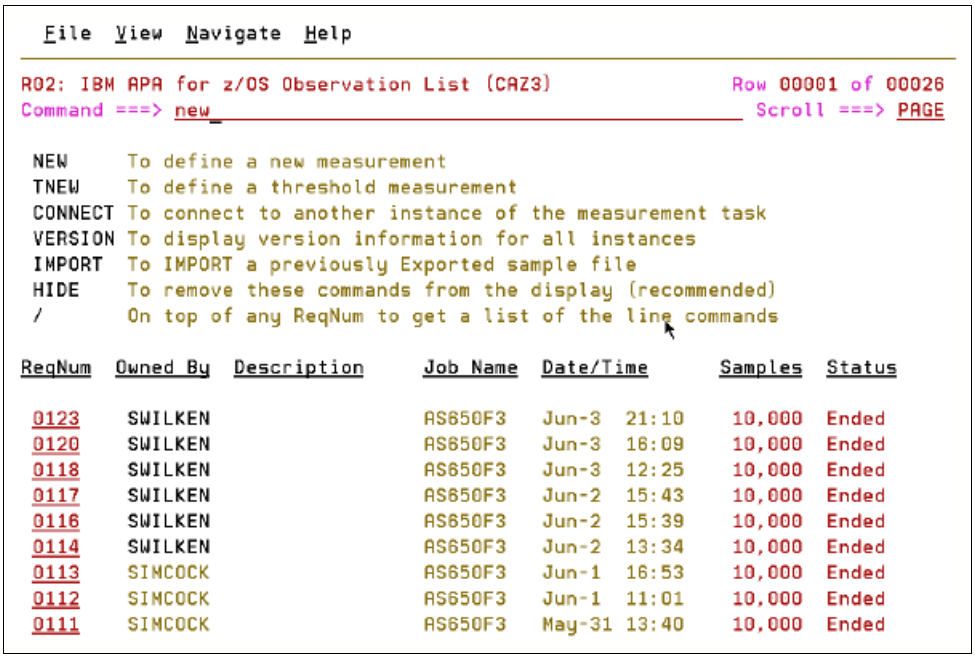


Figure 5-58 Create an observation

The Schedule New Measurement is displayed, as shown in Figure 5-59.

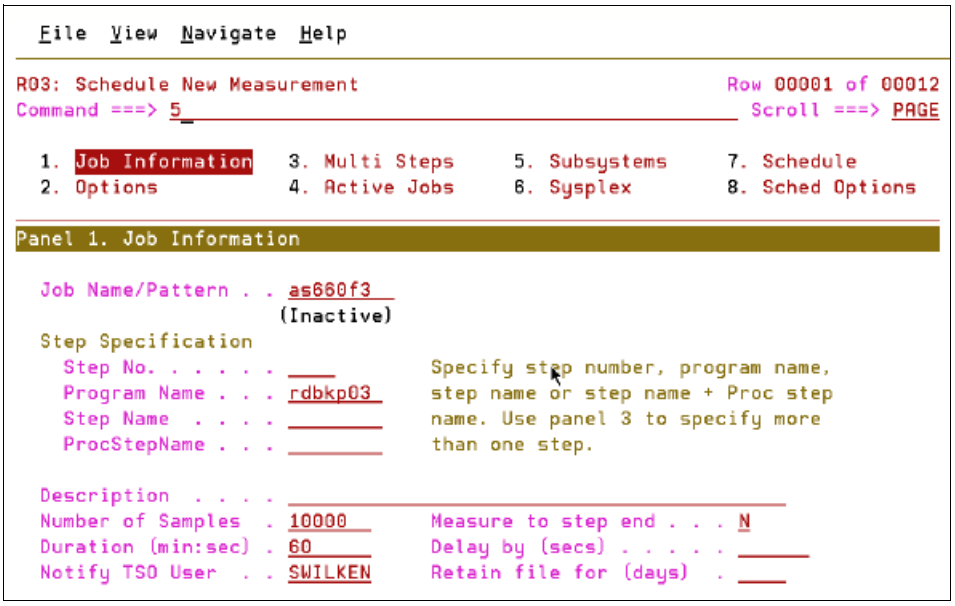


Figure 5-59 Schedule New Measurement display

3. Enter the following values:
   - Job name/Pattern field: AS660F3.
   - Program Name field: RDBKP03.
   - Number of Samples: 10000.
   - Duration (min:sec): 60
4. Select option **5 Subsystems** on the command line to display the CICS Transactions and Terminals display, as shown in Figure 5-60.

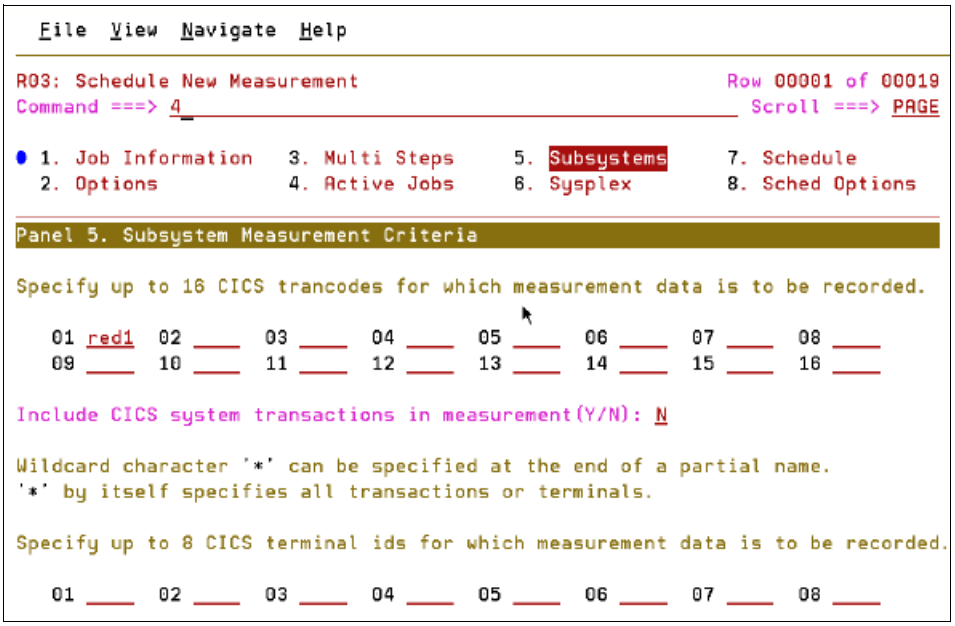


Figure 5-60 Select option 5

5. Enter the name of the CICS transaction in the **01** field and press Enter.

6. Select option **4 Active Jobs** and enter **S** against the job to be activated, as shown in Figure 5-61.

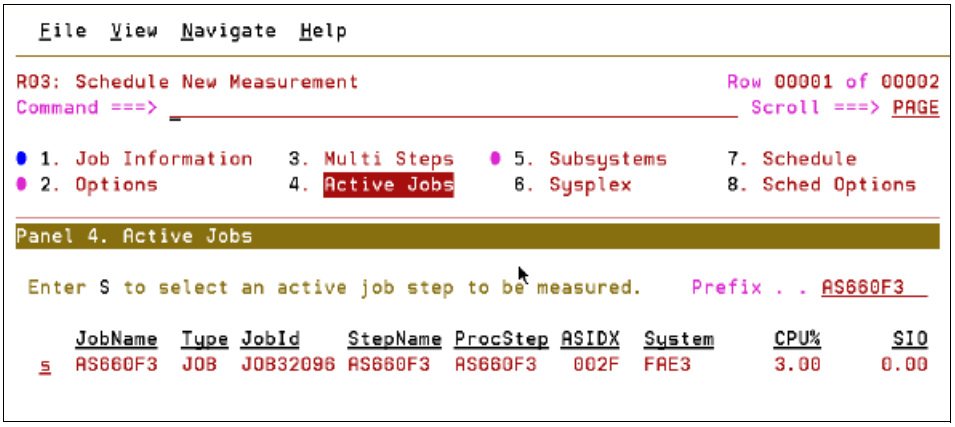


Figure 5-61 Select option for and enter s next to the jobname

After Enter is pressed, we see that our measurement is active, as shown in Figure 5-62.

Figure 5-62 Measurement is now active

7. Select option **1**, and press Enter without making any other changes,

   We return to the APA for z/OS Observation List display, where observation 125 is added and is active, as shown in Figure 5-63.

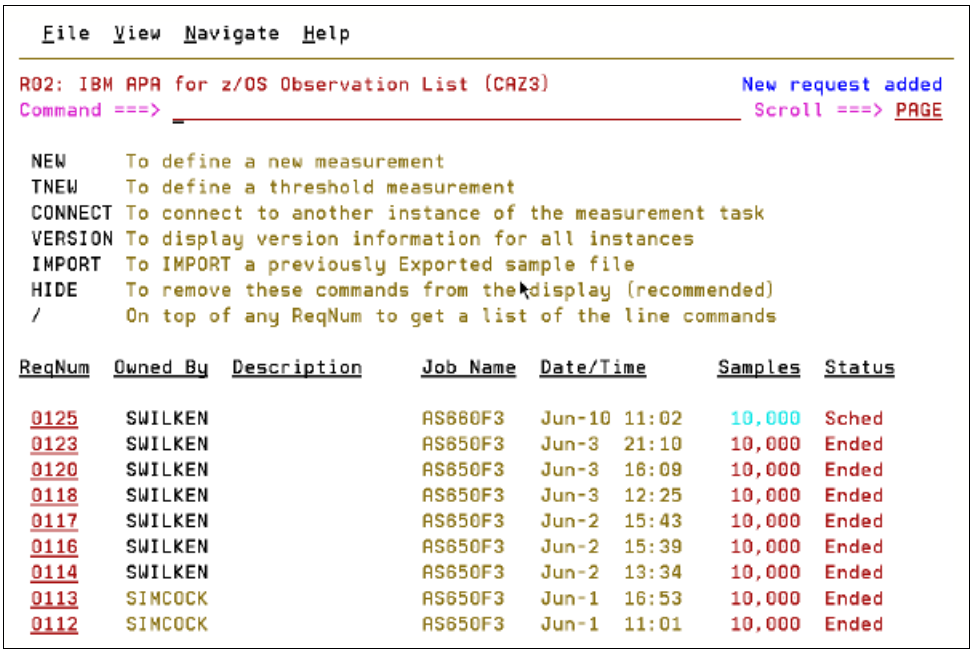


Figure 5-63 APA for z/OS Observation List display

We are now ready to start our looping CICS transaction.

8. When the analysis ends, enter S over the 0125 request number, as shown in Figure 5-64.

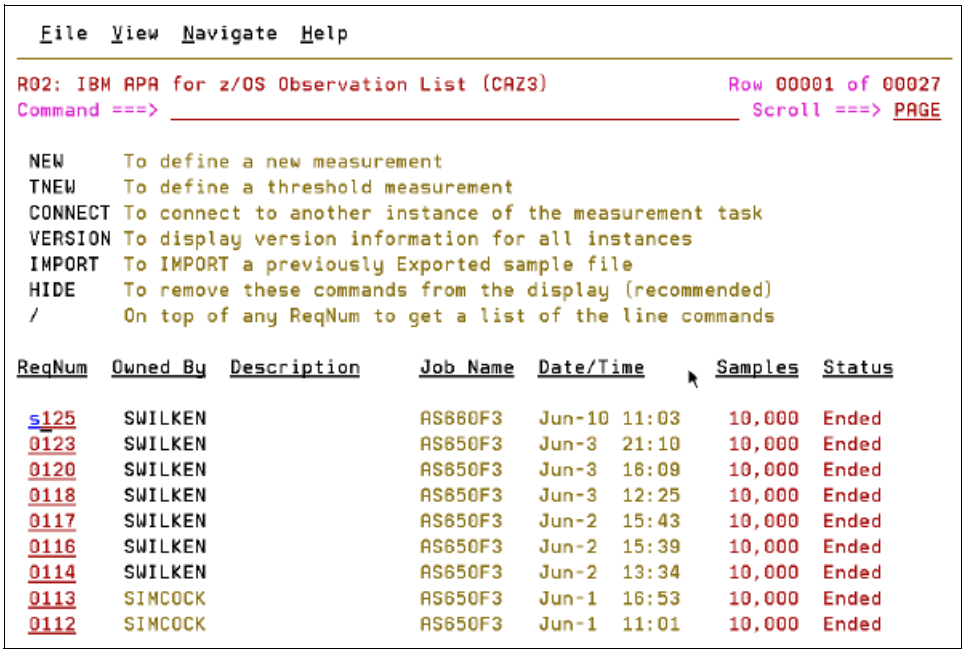


Figure 5-64 Type s over the 125 reqnum

The Performance Reports displays, as shown in Figure 5-65.

```
  File  View  Navigate  Help
R01: IBM APA for z/OS Performance Reports (0125/AS660F3)    Row 00001 of 00007
Command ===> c02_                                              Scroll ===> PAGE

  Select a category from     _ A Admin/Miscellaneous   _ I IMS Measurement
  the list to the right      _ S Statistics/Storage    _ E CICS Measurement
  to view the available      _ C CPU Usage Analysis    _ F DB2 Measurement
  reports in the selection   _ D DASD I/O Analysis     _ Q MQ Measurement
  list below.                _ W CPU WAIT Analysis     _ G Coupling Facility
                             _ H HFS Analysis          _ J Java Measurement
                             _ V Variance Reports      _ X Multi Address Space

                                                              More:     +
Enter S to make a selection or enter the report code on the command line

   _ S01 Measurement Profile                _ S07 TCB Execution Summary
   _ S02 Load Module Attributes             _ S08 Processor Utilization Summary
   _ S03 Load Module Summary                _ S09 Measurement Analysis
   _ S04 TCB Summary
   _ S05 Memory Usage Timeline
   _ S06 Data Space Usage Timeline
```

Figure 5-65   Performance Reports display

9. Enter C02 on the command line to display the CPU Usage by Module report, as shown in Figure 5-66.

```
  File  View  Navigate  Help
C02: CPU Usage by Module (0125/AS660F3)                     Row 00001 of 00024
Command ===> _                                                 Scroll ===> PAGE

Name       Description         Percent of CPU Time * 10.00%  ±1.4%
                                   *....1....2....3....4....5....6....7....8....9
RDBKP03    Application        99.17 =============================================
           Program
DFHSIP     CICS Services       0.18
CEECCICS   LE-CICS             0.12
           interface routine
DFHAIP     CICS Services       0.12
DFHTSDML   CICS Services       0.10
IDA019L1   Virtual I/O         0.08
           (VIO) and VSAM
IEAVELK    Supervisor          0.08
           Control
IGVVSTOR   Virtual storage     0.02
           management (VSM)
IAXVF      Real storage        0.02
           manager (RSM)
DFHEITS    Temporary           0.02
           storage exec
```

Figure 5-66   CPU Usage by Module report

We can see that the RDBKP03 module is responsible for using the most CPU time.

10. Typeover the first character (**R**) of the module name with a plus sign (+), and press Enter,

    We can see a further distribution of CPU time per CSECT, as shown in Figure 5-67.

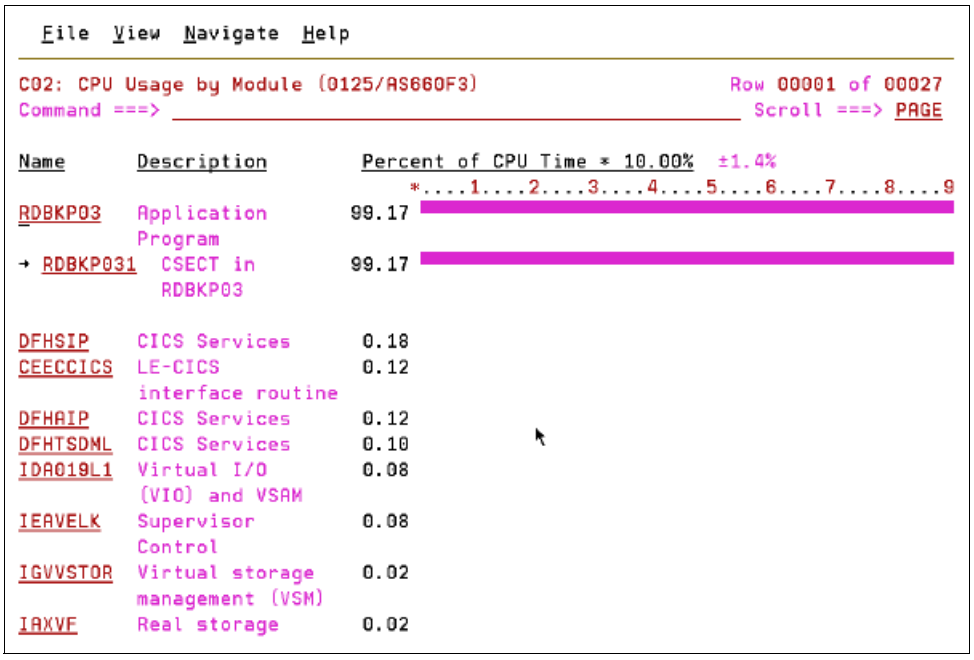


Figure 5-67 CPU time per CSECT

In our example, there is only one CSECT, RDBKP031.

To further identify the source line (or lines) that is responsible for using the most CPU time, we must provide the source mapping file for this program.

11. Enter `A01` on the command line, and press Enter.

12. The Source Program Mapping panel displays, as shown in Figure 5-68.

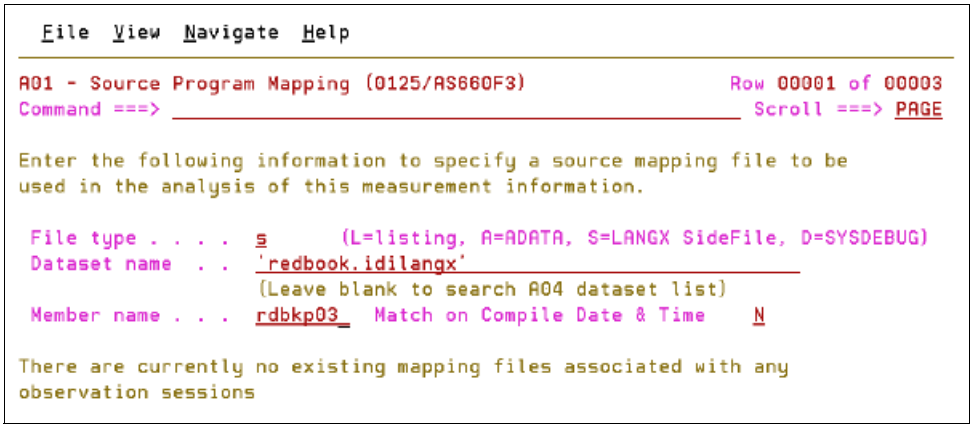


Figure 5-68 Source Program Mapping panel

13. Enter S as the file type (LANGX), 'REDBOOK.IDILANGX' as the data set name, and RDBKP03 as the member name. Press Enter.

    We now see that the specified source mapping file was loaded for our report, as shown in Figure 5-69.

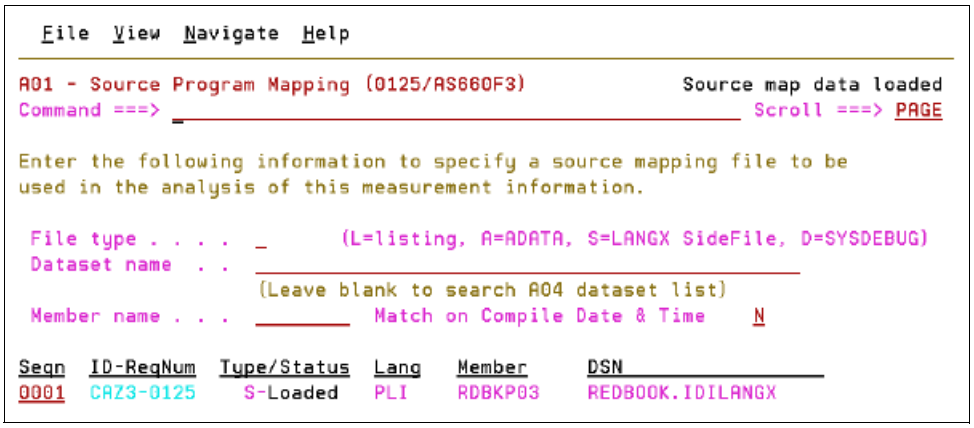


Figure 5-69 Source mapping file was loaded for our report

14. Press PF3 to return to the CPU usage report.

15. Typeover the first character of the CSECT name with a P to perform source mapping, as shown in Figure 5-70.

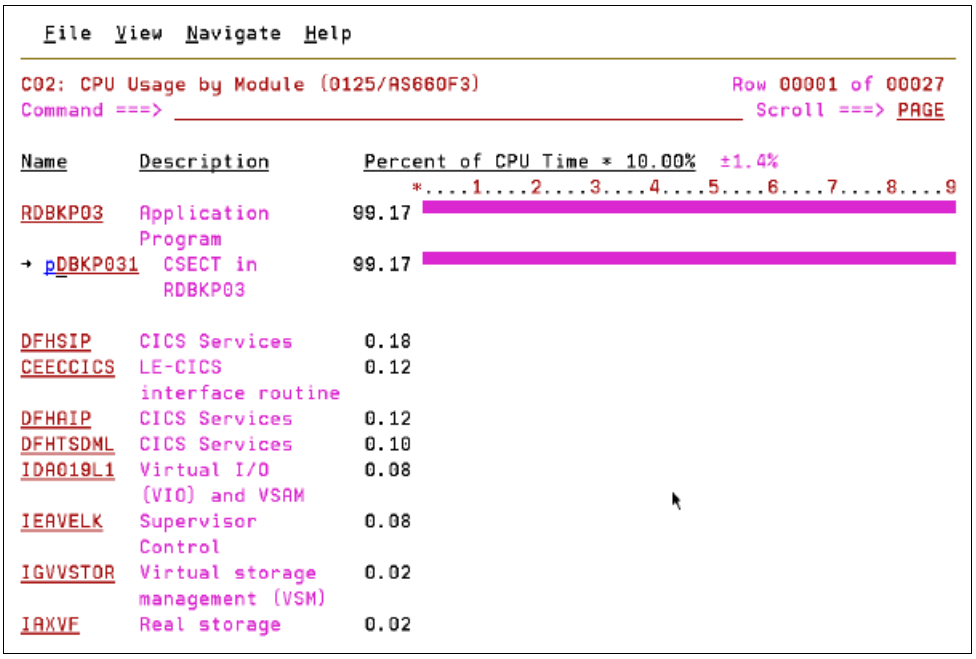


Figure 5-70 Enter p to perform source mapping

The Source Program Attribution panel is displayed, as shown in Figure 5-71.

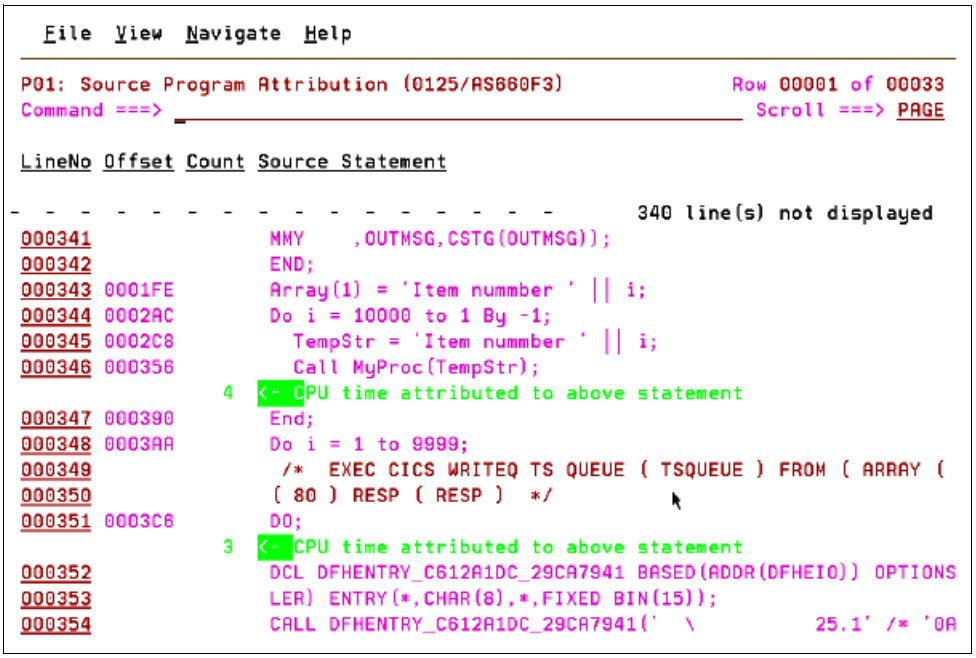


Figure 5-71 Source Program Attribution panel

Scrolling down a page, we clearly see the cause of our high CPU usage in this program, as shown in Figure 5-72.

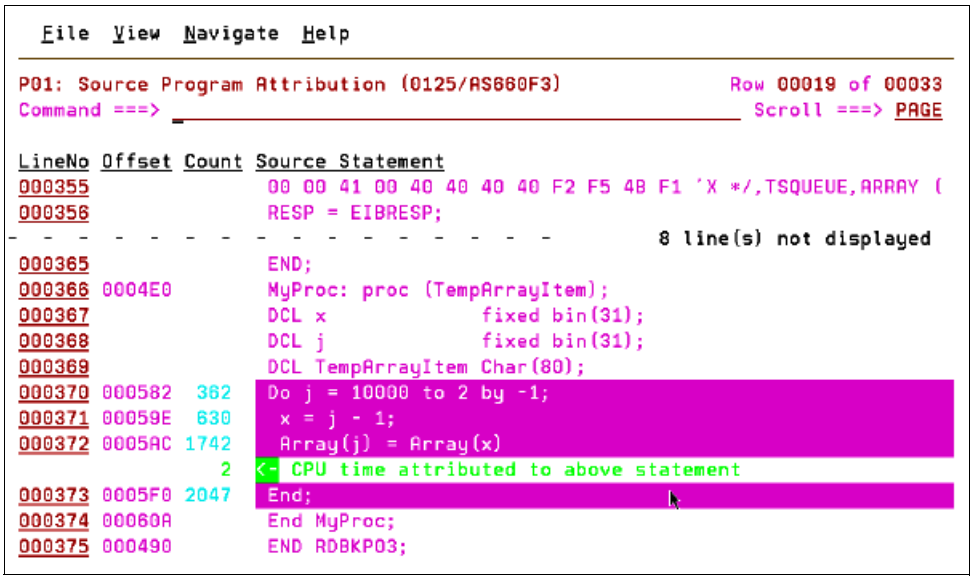


Figure 5-72 CPU usage

5.4.2 Starting Application Performance Analyzer from CICS Explorer

After the Application Performance Analyzer perspective is selected, a connection between the Application Performance Analyzer that is running on CICS Explorer and the host system in which our CICS region is running must be established.

Complete the following steps to establish this connection:

1. Click the **Connect APA** icon in the toolbar, as shown in Figure 5-73.

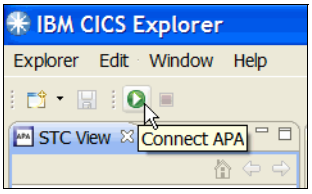


Figure 5-73 Connect APA

The Logon to z/OS window is displayed, as shown in Figure 5-74.

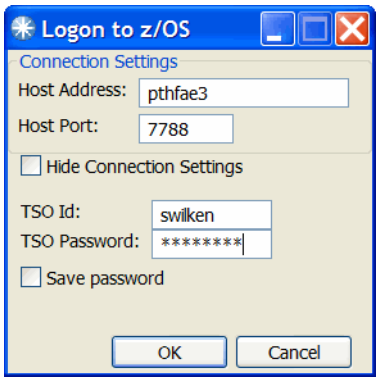


Figure 5-74 Log on to z/OS

2. Enter the host address, host port, TSO ID, and TSO Password, then click **OK**.

   The host port number must match the number that is used by the Application Performance Analyzer server job that is running on the host system. This number often is provided by your systems programmer.

3. Refresh the APA Observations List by clicking the **Refresh Observations List** icon in the toolbar, as shown in Figure 5-75.

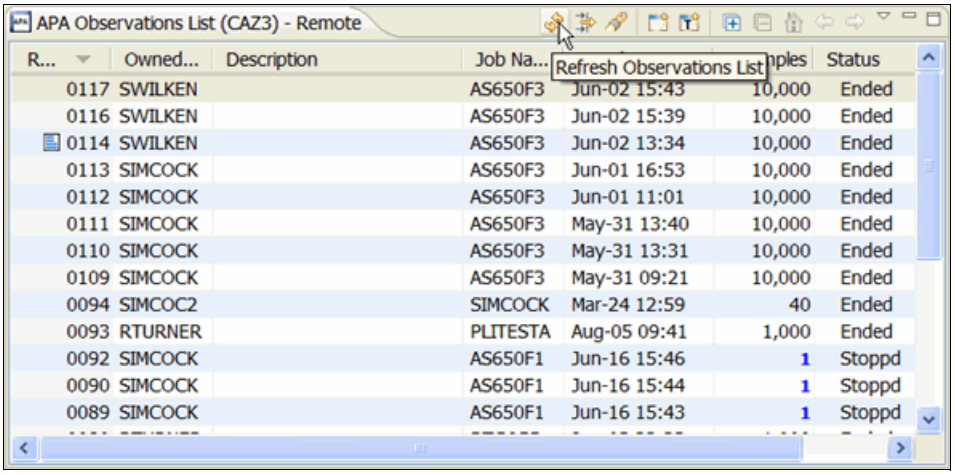


Figure 5-75 APA Observations List

4. To create an observation, click the **New Observation** icon in the toolbar, or right-click an existing entry and select **New** from the menu.

   The New Observation window is displayed, as shown in Figure 5-76.

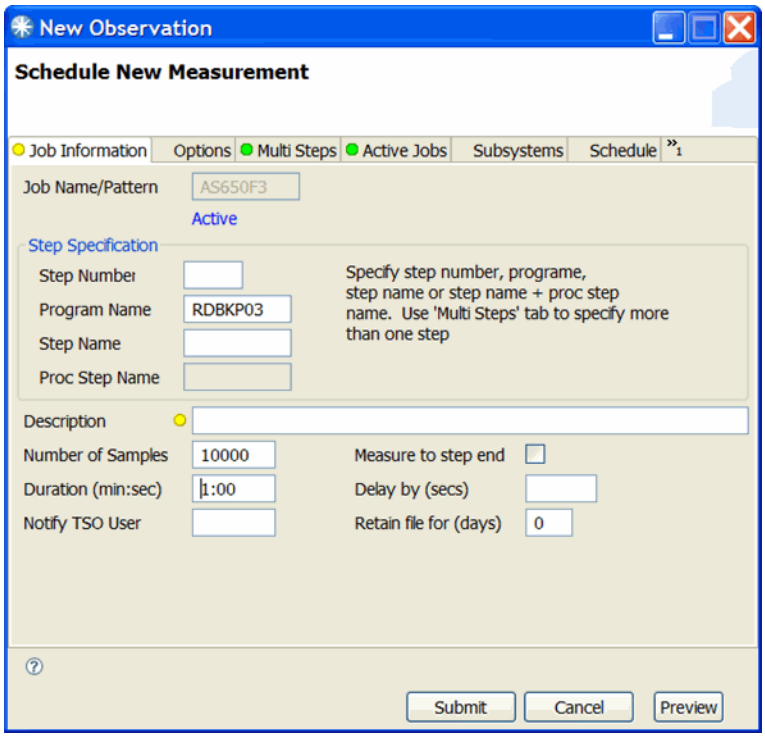


Figure 5-76 New Observation window

5. Enter the following information:

   – Job Name/Pattern: `AS650F3`
   – Program Name: `RDBKP03`
   – Number of Samples: `10000`
   – Duration: `1:00`

6. Click the **Subsystems** tab to display the CICS/IMS transactions details, as shown in Figure 5-77.

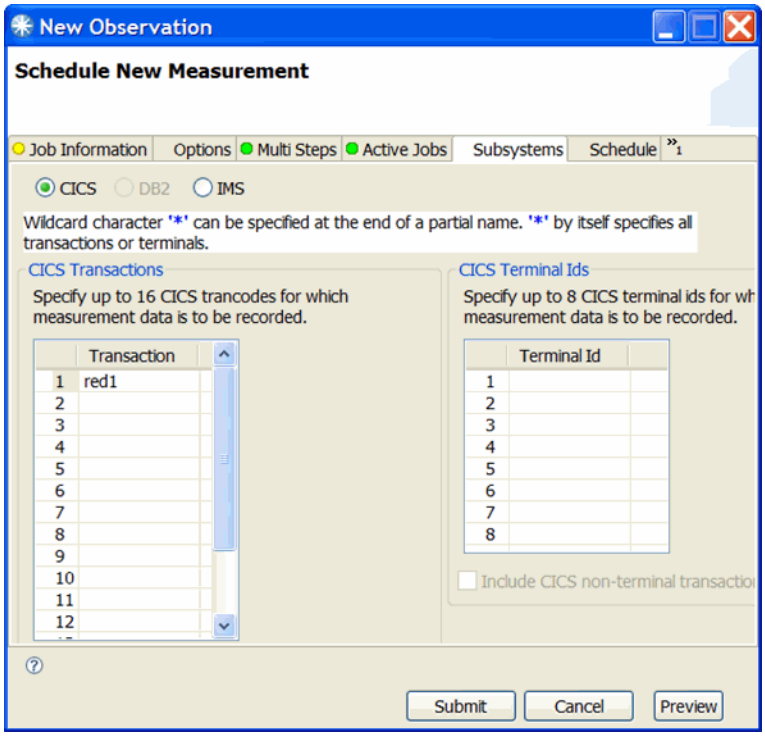


Figure 5-77 Subsystems tab

7. Ensure that CICS is selected and enter the transaction ID that is to be analyzed (in our case, `RED1`). Click **Submit**.

   Our new observation, 120, is added to the APA Observations List, as shown in Figure 5-78.

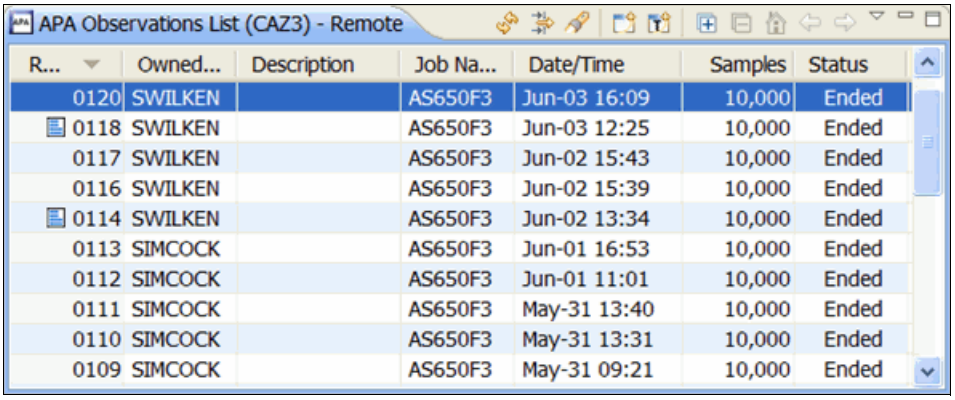


Figure 5-78 APA Observations List

We are now ready to start our looping CICS transaction.

8. When the analysis ends, right-click the observation and select **Download Reports**. The Downloading Reports window is displayed, as shown in Figure 5-79.

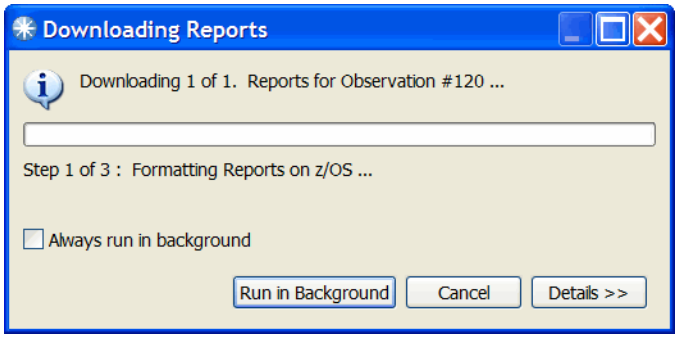


Figure 5-79 Download reports

The reports are then presented in the Report View, as shown in Figure 5-80.

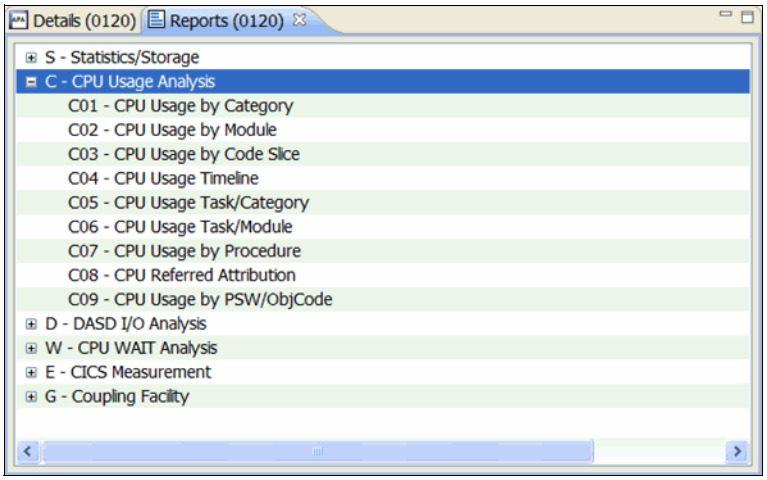


Figure 5-80 Report View

One of the reports that is available is the CPU Usage by Module (C02) report.

9. As shown in Figure 5-81, selecting C02 tells us how significant a percentage of the CPU time was spent in which module.

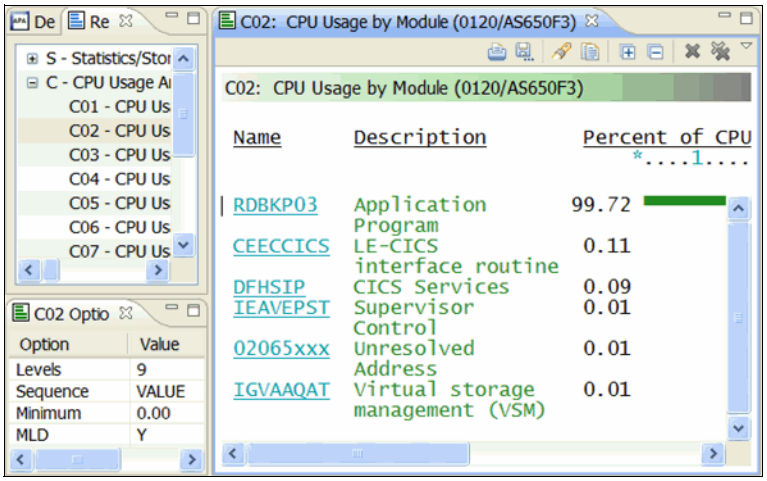


Figure 5-81 CPU Usage by Module

We can see that most of the CPU time was spent in module RDBKP03.

10. Click the RDBKP03 module name to see the CSECTs in the module and how much CPU time was spent in each module.

As shown in Figure 5-82, our module contains only one CSECT named RDBKP031.

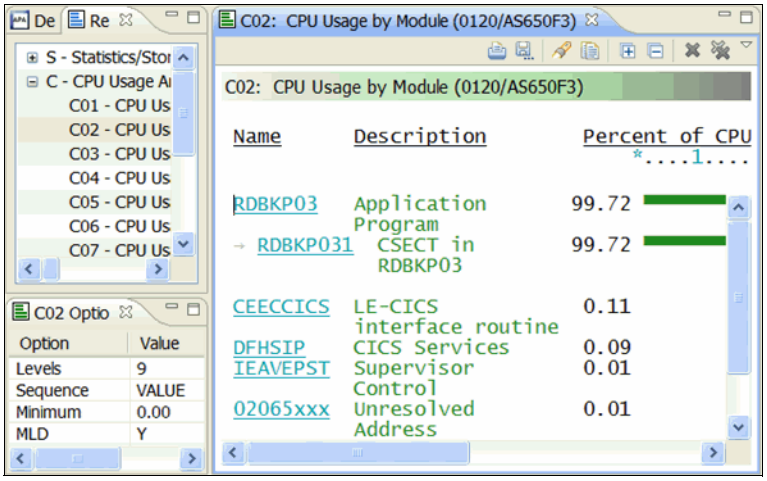


Figure 5-82 CSECTs in the module and CPU time that is spent in each one

5.5 Starting Debug Tool by using container field information

The test program that is used in this scenario consists of a COBOL program (CDAT1), which calculates birthdate information (CDAT2), or retirement information (CDAT3).

5.5.1 Debug Tool sessions

We use the birthdate transaction to demonstrate how you can capture a Debug Tool session for an instance of a transaction. The Debug Tool session can be displayed on a CICS terminal, or in the GUI.

5.5.2 Starting a CDAT transaction

Complete the following steps to start CDAT and demonstrate what displays on a normal run:

1. Log on to CICS. Clear the window and enter CDAT, as shown in Figure 5-83. Press Enter.

```
cdat
```

Figure 5-83 CDAT transaction

2. Select B for birthday calculation, and enter a date, as shown in Figure 5-84. Press Enter.

```
 ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


    19830505   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

       B   <=== ENTER REQUEST
            B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
            R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
            C : CLEAR AND START OVER
            @ : ABEND WITH SOC7









F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-84 Selecting option B

3. The birthday results are displayed, as shown in Figure 5-85.

```
  ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


     19830505   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

        B    <=== ENTER REQUEST
              B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
              R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
              C : CLEAR AND START OVER
              @ : ABEND WITH S0C7

HERE IS YOUR BIRTHDATE AND # OF DAYS ELAPSED

YOUR BIRTHDATE AND DAY: Thursday 05 May 1983
HOW LONG AGO WAS THIS?        10,400 DAYS



F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-85 Birthday results panel

4. Set up the Debug Tool trap to capture the CDAT transaction, as shown in Figure 5-86.

```
DTCN               Debug Tool CICS Control - Primary Menu              CICSACB1
                          * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id      ==> 0047
 Transaction Id   ==>
 LoadMod::>CU(s) ==> CDAT*     ::> CDAT*          ==>          ::>
                 ==>           ::>                ==>          ::>
                 ==>           ::>                ==>          ::>
                 ==>           ::>                ==>          ::>
 User Id          ==> DNET845
 NetName          ==>
 IP Name/Address ==>
Select type and ID of debug display device
 Session Type     ==> MFI                  MFI, TCP
 Port Number      ==>                      TCP Port
 Display Id       ==> 0047

 Generated String:  TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Repository String: TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Profile Status:    Active. Press PF5 to Inactivate.
 EQA2514I Debug Tool profile saved
 PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
```

Figure 5-86 Debug setup to capture the CDAT transaction

5. Press F3 to exit. Enter CDAT and press Enter. The Debug Tool session starts when the CDAT transaction runs, as shown in Figure 5-87 on page 203.

```
COBOL    LOCATION: CDAT1 ENTRY
Command ===>                                                Scroll ===> HALF
MONITOR -+-----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 3
******************************* TOP OF MONITOR *******************************
                                  ----+----1----+----2----+----3----+----4----
0001   1 W-COM-USER-REQUEST          '.'
0002     ********** AUTOMONITOR **********
0003     There are no variables in the statement to display.
****************************** BOTTOM OF MONITOR ******************************
SOURCE: CDAT1 ----1----+----2----+----3----+----4----+----5----+ LINE: 2 OF 576
        2         PROGRAM-ID. CDAT1.                                          .
        3       *    CICS / DEBUG TOOL DEMO PROGRAM                           .
        4       *                                                             .
        5       *    THIS PROGRAM WILL RECEIVE A DATE AND COVERT THE DATE TO  .
        6       *    AN INTEGER IN A CALLED PROGRAM TO DETERMINE DAYS FROM    .
        7       *    CURRENT DATE.                                            .
        8       *                                                             .
        9       *    (C) 2004 IBM - KEVIN J. CUMMINGS RESERVED.               .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 28 OF 30
0028   SET DEFAULT SCROLL HALF ;
0029   SET TEST ERROR ;
0030  *** User preferences file commands end ***
PF   1:CURLINE    2:STEP       3:QUIT       4:MONITOR    5:FIND       6:AT/CLEAR
PF   7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 5-87 Debug starting when CDAT entered

6. We can now step through the program. When user input is needed, the CDAT panel is displayed. Enter the data on the panel and press Enter, as shown in Figure 5-88.

```
 ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


    19830505   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

       B   <=== ENTER REQUEST
             B : SEE YOUR BIRTHDAY            (LINK TO PROGRAM CDAT2)
             R : CALCULATE RETIREMENT            (CALL PROGRAM CDAT3)
             C : CLEAR AND START OVER
             @ : ABEND WITH S0C7









F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-88 Enter option B

7. The information that we entered is passed to our Debug Tool session, as shown in Figure 5-89. We can continue to step through the program.

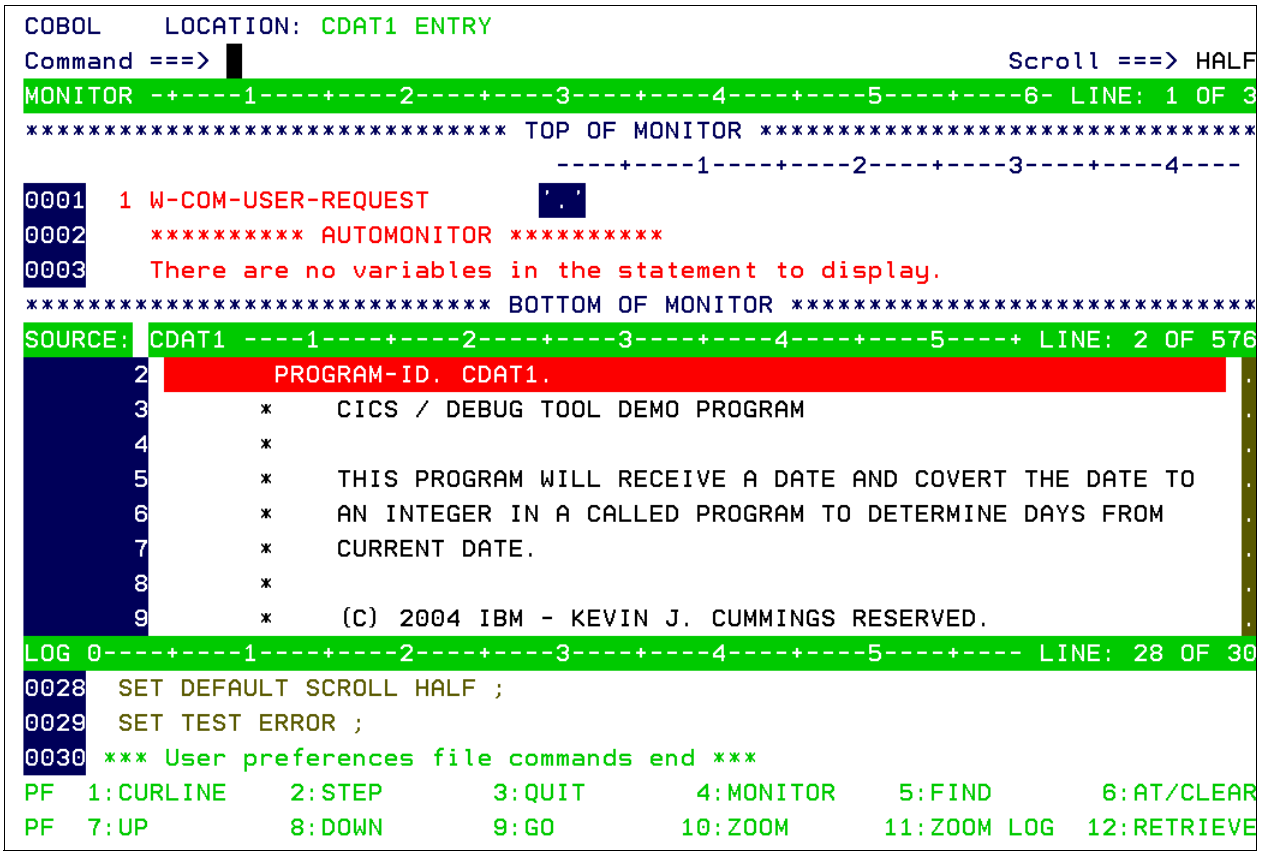


Figure 5-89 We enter back into Debug tool

8. We now change the CICS trap to remove the input Terminal ID and User ID, as shown in Figure 5-90.

```
DTCN               Debug Tool CICS Control - Primary Menu              CICSACB1
                          * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id      ==>
 Transaction Id   ==>
 LoadMod::>CU(s)  ==> CDAT*     ::> CDAT*          ==>          ::>
                  ==>           ::>                ==>          ::>
                  ==>           ::>                ==>          ::>
                  ==>           ::>                ==>          ::>
 User Id          ==>
 NetName          ==>
 IP Name/Address  ==>
Select type and ID of debug display device
 Session Type     ==> MFI                  MFI, TCP
 Port Number      ==>                      TCP Port
 Display Id       ==> 0047

 Generated String:  TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Repository String: TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Profile Status:    Active. Press PF5 to Inactivate.
 EQA2514I Debug Tool profile saved
 PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
```

Figure 5-90 Removing terminal and user IDs

9. Use PF8 Adv for advanced selection. We want to trap the birthday 19830505. Press F3 to return, as shown in Figure 5-91.

```
DTCN              Debug Tool CICS Control - Advanced Options         CICSACB1

Select advanced program interruption criteria:

Commarea Offset  ==> 0
Commarea Data    ==> 19830505

Container Name   ==>
Container Offset ==> 0
Container Data   ==>

URM Debugging    ==> NO









Default offset and data representation is decimal/character.
See Help for more information.



PF1=HELP 2=GHELP 3=RETURN
```

Figure 5-91 Set trap to birthday of 19839506

10. After you return to the main entry panel, press F4 to Save the updates. Press F3 to exit, as shown in Figure 5-92.

```
DTCN               Debug Tool CICS Control - Primary Menu              CICSACB1
                           * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id      ==>
 Transaction Id   ==>
 LoadMod::>CU(s)  ==> CDAT*     ::> CDAT*          ==>          ::>
                  ==>           ::>                ==>          ::>
                  ==>           ::>                ==>          ::>
                  ==>           ::>                ==>          ::>
 User Id          ==>
 NetName          ==>
 IP Name/Address  ==>
Select type and ID of debug display device
 Session Type     ==> MFI                  MFI, TCP
 Port Number      ==>                      TCP Port
 Display Id       ==> 0047

 Generated String:  TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Repository String: TEST(ERROR,'*',PROMPT,'MFI%0047:DNET845.ADLAB.FILES(CICPREF)
')
 Profile Status:    Active. Press PF5 to Inactivate.
 EQA2514I Debug Tool profile saved
 PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
```

Figure 5-92 After PF3 pressed

11. Enter CDAT. Debug Tool does not start because it is waiting for the data in the COMMAREA to match what we entered. Enter a different birthday, as shown in Figure 5-93.

```
  ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


     19600401   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

        B   <=== ENTER REQUEST
             B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
             R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
             C : CLEAR AND START OVER
             @ : ABEND WITH S0C7









F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-93 Waits to match birthday we entered

12. Press Enter to process the date. The results are displayed, as shown in Figure 5-94.

```
  ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


     19600401   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

        B   <=== ENTER REQUEST
             B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
             R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
             C : CLEAR AND START OVER
             @ : ABEND WITH S0C7

HERE IS YOUR BIRTHDATE AND # OF DAYS ELAPSED

YOUR BIRTHDATE AND DAY: Friday 01 April 1960
HOW LONG AGO WAS THIS?        18,834 DAYS



F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-94 Results displayed

13. Change the birthday to the data that we entered in to the Advanced Options in the trap, as shown in Figure 5-95. Press Enter.

```
  ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION


     19830505   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

        B   <=== ENTER REQUEST
             B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
             R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
             C : CLEAR AND START OVER
             @ : ABEND WITH S0C7

HERE IS YOUR BIRTHDATE AND # OF DAYS ELAPSED

YOUR BIRTHDATE AND DAY: Friday 01 April 1960
HOW LONG AGO WAS THIS?        18,834 DAYS



F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

Figure 5-95 Enter date that we wanted to trap

14. Debug Tool triggers. The program that we stopped in was CDAT2, which is the birthday calculation program. The COMMAREA is displayed because we added it to our Monitor window. See Figure 5-96.

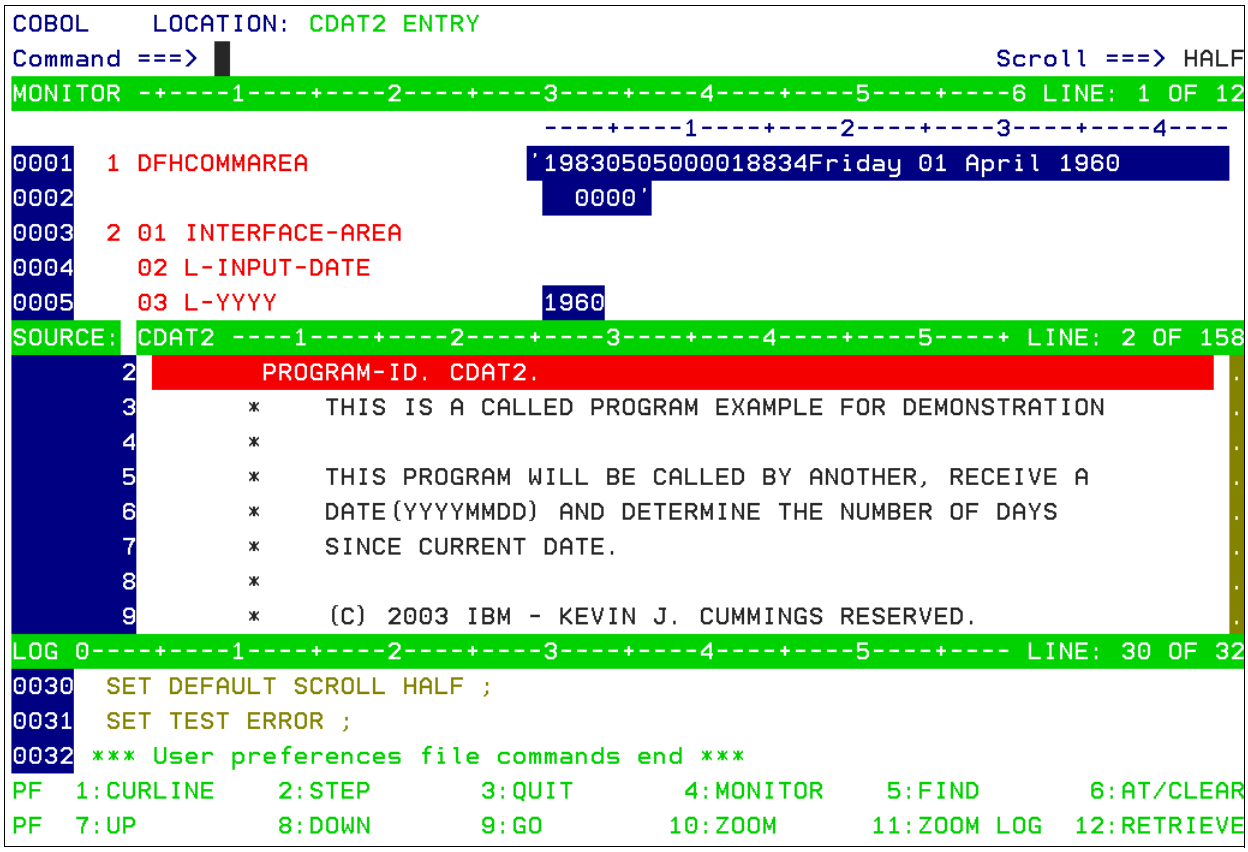


Figure 5-96 We now enter debug mode

This method of debugging can be useful when debugging transactions that are started from a web application. In that case, there is no terminal ID or user ID to match on. By using this method, we avoid stopping programs in Debug Tool unintentionally, and possibly affecting other users.

6

Using IBM Problem Determination Tools with Batch

The scenario that we use in this chapter involves an abend in a COBOL program, which is running as part of a batch job. The chapter includes descriptions of how Fault Analyzer, Debug Tool, and File Manager can be used to analyze and correct the problem. It also includes a description of how Application Performance Analyzer can be used for analyzing problem areas in batch programs.

Where applicable, we describe the use of the tools from a z/OS SDSF or TSO/ISPF session, and from CICS Explorer on a workstation.

6.1 Using Fault Analyzer with a batch program

In this section, we describe various ways to use Fault Analyzer.

6.1.1 Starting Fault Analyzer from a batch job

Figure 6-1 shows the JCL that is used to submit the job that is described in this scenario. There are some Fault Analyzer Data Definitions (DDs) in the example for reference. Normally, Fault Analyzer does not require any JCL updates so that it can be started. The running program, SAM1, processes two input files and produces a report. It calls two other programs, SAM2 and SAM3, as part of this process.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.ADLAB.JCL(XSAMFA) - 01.04               Columns 00001 00072
Command ===> sub                                          Scroll ===> CSR
000007 //*    PROGRAMS THAT ARE COMPILED BUT THE SYSDEBUG FILE WAS NOT SAVED
000008 //*    TO PREVENT FAULT ANALYZER FROM AUTOMATICALLY FINDING SOURCE INFO
000009 //*****************************************************************
000010 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000011 //STEPLIB  DD DSN=&SYSUID..ADLAB.LOAD,DISP=SHR
000012 //CUSTFILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000013 //SYSPRINT DD SYSOUT=*
000014 //SYSOUT   DD SYSOUT=*
000015 //CUSTRPT  DD SYSOUT=*
000016 //CUSTOUT  DD SYSOUT=*
000017 //TRANFILE DD *
000018 *TRAN    (*  IN COL 1 IS A COMMENT)
000019 *-----
000020 PRINT      <== PRINT CUSTOMER LIST
000021 XXXX       BAD TRANSACTION
000022 TOTALS     <== PRINT TOTALS
000023 //*  ABEND      <== WILL CAUSE DIVIDE BY ZERO ABEND
000024 //*
000025 //*****************************************************************
000026 //*    SAMPLE OPTIONAL FILES FOR FAULT ANALYZER:
000027 //*****************************************************************
000028 //IDIHIST  DD DISP=SHR,DSN=TSS12.FAULT.HISTORY
000029 //*     IDIOPTS CAN BE USED TO SPECIFY FAULT ANALYZER PARAMETERS
000030 //IDIOPTS  DD *
000031  INCLUDE,MAXMINIDUMPPAGES(1000)
000032  NODUP(NORMAL(0))
000033 /*
000034 //*     IDIOFF, IF PRESENT, WILL TURN OFF FAULT ANALYSIS FOR THE STEP
000035 //*        //IDIOFF   DD DUMMY
000036 //*     IDILCOB IS A PDS CONCATENATION FOR COBOL COMPILER LISTINGS:
000037 //*        //IDILCOB  DD DSN=&SYSUID..ADLAB.LISTING,DISP=OLD
000038 //*     IDISYSDB IS A PDS CONCATENATION FOR SYSDEBUG FILES:
000039 //*        //IDISYSDB DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=OLD
000040 //*     IDITRACE CAN BE USED TO PRINT A TRACE OF THE LISTINGS SEARCH
000041 //IDITRACE DD SYSOUT=*
****** ***************************** Bottom of Data ****************************
```

Figure 6-1 JCL used

The IDIHIST DD specifies the name of the history file to which the fault report is written. The IDIOPTS DD includes an override for the MAXMINIDUMPAGES. The NODUP(NORMAL (0) overrides the duplicates window that allows this job to be added to the history file even if there was a duplicate entry added before this entry. The XSAMFA member is used in the PD Tools Mentor Workshop training.

In our example, the job fails, which results in an 0C7 abend, as shown in Figure 6-2.

```
 09.39.29 JOB02474 $HASP165 TSS12X   ENDED AT STLABF7 - ABENDED S0C7 U0000 CN(IN
TERNAL)
 ***
```

Figure 6-2 Job fails

The SDSF output reveals that Fault Analyzer was started, and that Fault ID 00016 was written to the history file that was specified in the JCL, as shown in Figure 6-3.

```
  Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TSS12X   JOB02474  DSID     2 LINE 0       COLUMNS 21- 100
 COMMAND INPUT ===> _                                        SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
 J E S 2  J O B  L O G  --  S Y S T E M  F 6      --  N O D E  S T L A B F 7

--- FRIDAY,     30 SEP 2011 ----
IRR010I  USERID TSS12     IS ASSIGNED TO THIS JOB.
ICH70001I TSS12     LAST ACCESS AT 09:37:06 ON FRIDAY, SEPTEMBER 30, 2011
$HASP373 TSS12X   STARTED - INIT 12   - CLASS A - SYS F6
+IDI0001I Fault Analyzer V11R1M0 (UK65276 2011/03/01) invoked by IDIXCEE using S
+IDI0002I Module SAM2, program SAM2, source line # 89: Abend S0C7 (Data Exceptio
+IDI0003I Fault ID F00016 assigned in history file TSS12.FAULT.HISTORY
IEF450I TSS12X RUNSAM1 - ABEND=S0C7 U0000 REASON=00000007
SMF000I  TSS12X       RUNSAM1      SAM1         00C7
$HASP395 TSS12X   ENDED
TISTICS ------
XECUTION DATE
 READ
T PRINT RECORDS
T PUNCH RECORDS
T SPOOL KBYTES
ES EXECUTION TIME
******************************** BOTTOM OF DATA ********************************
```

Figure 6-3 JSDSF output that shows Fault Analyzer was started

Because the Fault Analyzer option DEFERREDREPORT was not used, the analysis report is written to the IDIREPRT DD, as shown in Figure 6-4.

```
  Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB TSS12X   (JOB02474)     DATA SET DISPLAYED
COMMAND INPUT ===>                                        SCROLL ===> CSR
NP   DDNAME   StepName ProcStep DSID Owner    C Dest              Rec-Cnt Page
     JESJCLIN                    1 TSS12    H                        33
     JESMSGLG JES2               2 TSS12    H LOCAL                  19
     JESJCL   JES2               3 TSS12    H LOCAL                  37
     JESYSMSG JES2               4 TSS12    H LOCAL                 101
     $INTTEXT JES2               5 TSS12    A                        12
     TRANFILE RUNSAM1          101 TSS12    H                         5
     IDIOPTS  RUNSAM1          102 TSS12    H                         2
     SYSOUT   RUNSAM1          104 TSS12    H LOCAL                   5
     CUSTRPT  RUNSAM1          105 TSS12    H LOCAL                  36
     IDITRACE RUNSAM1          107 TSS12    H LOCAL                 376
     CEEDUMP                   108 TSS12    H LOCAL                 764
s    IDIREPRT RUNSAM1          109 TSS12    H LOCAL               1,214
```

Figure 6-4 IDIREPRT DD

The Fault Analyzer report is written to IDIREPRT, which contains details about the abend that occurred, as shown in Figure 6-5.

```
  Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TSS12X   JOB02474  DSID   109 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
*******************************************************************************
* IBM Fault Analyzer for z/OS V11R1M0 (UK65276 2011/03/01)                    *
*                                                                             *
*     Copyright IBM Corp. 2000, 2011.  All rights reserved.                   *
*******************************************************************************

JOBNAME: TSS12X    SYSTEM ABEND: 0C7              STLABF6   2011/09/30  09:39:27


<H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0C7 occurred in module SAM2 program SAM2 at offset X'36C'.

A program-interruption code 0007 (Data Exception) is associated with this abend
and indicates that:

  A decimal digit or sign was invalid.

The cause of the failure was program SAM2 in module SAM2.  The COBOL source code
that immediately preceded the failure was:

  Source
  Line #
  ------
  000088        *     *** Add this customer's BALANCE to the grand total ***
  000089              COMPUTE BALANCE-TOTAL =
  000090                 BALANCE-TOTAL + CUST-ACCT-BALANCE

The COBOL source code for data fields involved in the failure:

  Source
  Line #
  ------
  000059              05  CUST-ACCT-BALANCE      PIC S9(7)V99  COMP-3.
```

Figure 6-5 Fault Analyzer report that shows ABEND 0C7

If we start Fault Analyzer under TSO/ISPF, and enter the name of the history file that was used, we can complete the following steps:

1. As shown in Figure 6-6, use the **I** command to start Interactive Re-Analysis. This process demonstrates the point-and-shoot navigation and shows how you can access File Manager from within Fault Analyzer.

```
  File  Options  View  Services  Help
------------------------------------------------------------------------------
IBM Fault Analyzer - Fault Entry List                      Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR

Fault History File or View  : 'TSS12.FAULT.HISTORY'

{The following line commands are available: ? (Query), V or S (View saved
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history), C (Copy fault entry), M (Move fault entry), X (XMIT fault
entry).}

   Fault_ID  Job/Tran  User_ID   Sys/Job  Abend   Date        Time      Dups  Progra
i_ F00017    TSS12X    TSS12     STLABF6  S0C7    2011/10/05  07:38:10        SAM2
__ F00016    TSS12X    TSS12     STLABF6  S0C7    2011/09/30  09:39:27        SAM2
__ F00001    TSS12X    TSS12     STLABF6  SNAP    2011/04/21  08:06:54        SAM1
__ F00319    TSS12X    TSS12     STLABF6  S0C7    2011/03/17  12:09:16        SAM2
__ F00003    CICSCI1C  CICPROD   STLABF3  AP0001  2004/06/03  11:33:43        n/a

**  Bottom of data.
```

Figure 6-6 Start Interactive Re-Analysis

2. Move to the yellow highlighted Option 3, as shown in Figure 6-7, and press Enter.

```
  File  View  Services  Help
Interactive Reanalysis Report                                  Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
JOBNAME: TSS12X     SYSTEM ABEND: 0C7          STLABF6    2011/10/05  07:38:10

Fault Summary:
Module SAM2, program SAM2, source line # 89 : Abend S0C7 (Data Exception).

Select one of the following options to access further fault information:
  1. Synopsis
  2. Event Summary
  3. Open Files
  4. Storage Areas
  5. Messages
  6. Language Environment Heap Analysis
  7. Abend Job Information
  8. Fault Analyzer Options

{Fault Analyzer maximum storage allocated: 2.22 megabytes.}

*** Bottom of data.
```

Figure 6-7 Option 3

3. Move to the yellow highlighted CUSTFILE File Name, as shown in Figure 6-8, and press Enter.

```
  File  View  Services  Help
System-Wide Open Files                                         Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
JOBNAME: TSS12X     SYSTEM ABEND: 0C7          STLABF6    2011/10/05  07:38:10


Event 1 Program SAM1 Open Files

File Name . . . . . . . . . : CUSTFILE
File Name . . . . . . . . . : CUSTRPT
File Name . . . . . . . . . : TRANFILE

Non-Event-Related Open Files

File Name . . . . . . . . . : CEEDUMP
File Name . . . . . . . . . : SYSOUT

*** Bottom of data.
```

Figure 6-8 Select CUSTFILE

4. Move to the yellow highlighted Data Set Name, as shown in Figure 6-9, and press Enter.

```
  File  View  Services  Help
File Information                                               Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
JOBNAME: TSS12X     SYSTEM ABEND: 0C7          STLABF6    2011/10/05  07:38:10

File Name . . . . . . . . . : CUSTFILE
  Data Set Name . . . . . . : TSS12.ADLAB.FILES(CUST2FA)
  File Attributes . . . . . : ORGANIZATION=SEQUENTIAL, ACCESS MODE=SEQUENTIAL,
                              RECFM=FIXED
  Last I/O Function . . . . : READ
  Open Status . . . . . . . : INPUT
  File Status Code. . . . . : 0
```

Figure 6-9 Select the data set name

5. As shown in Figure 6-10, select Edit to edit the data set with File Manager and use the associated COBOL copybook for a formatted display.

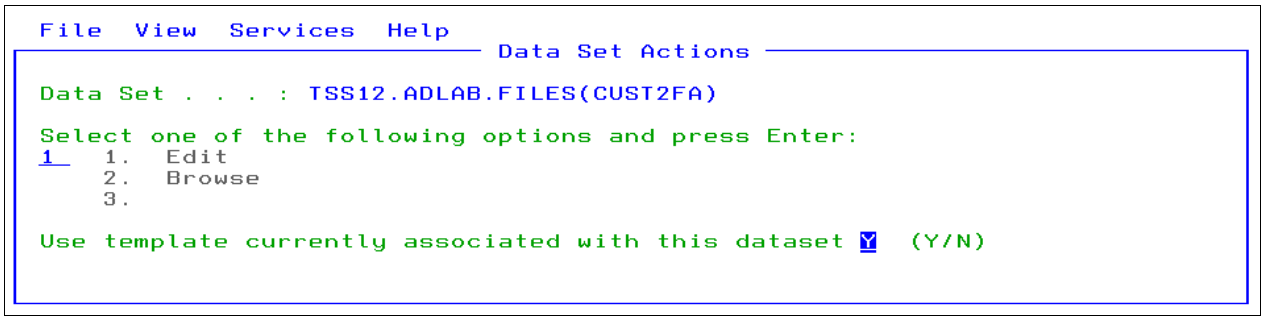


Figure 6-10 Start File Manager

6. From an edit session, use the Find Error (FE) command, as shown in Figure 6-11, to position to the bad data.

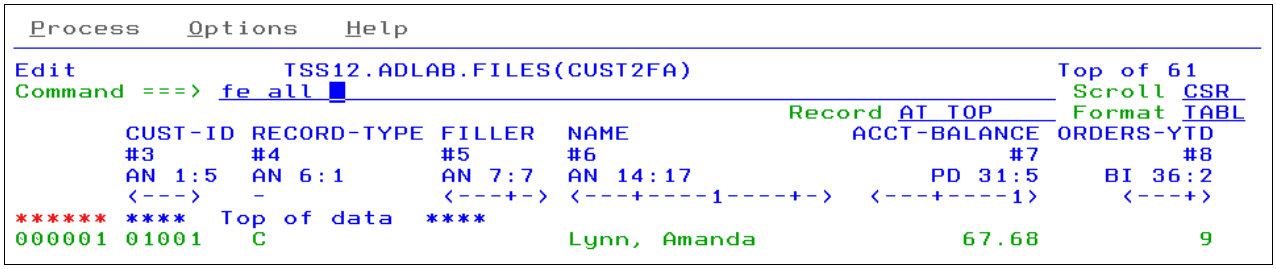


Figure 6-11 FE command

In Figure 6-12, the bad data was found and positioned to with a count of the total number of occurrences. The data can now be corrected.

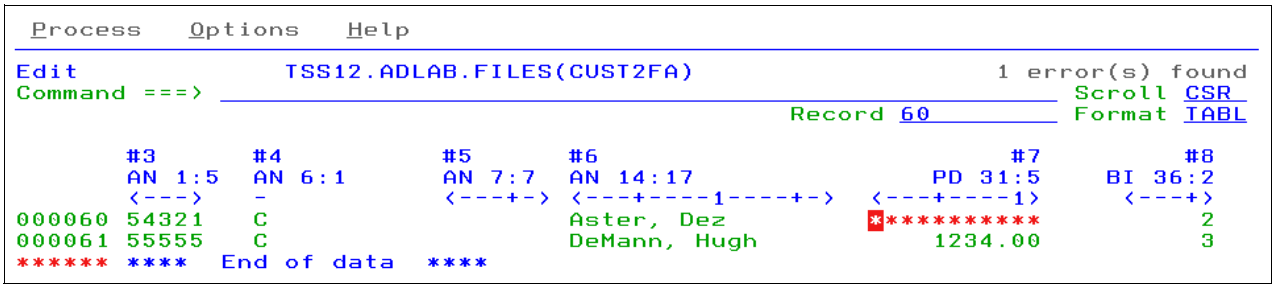


Figure 6-12 Error found

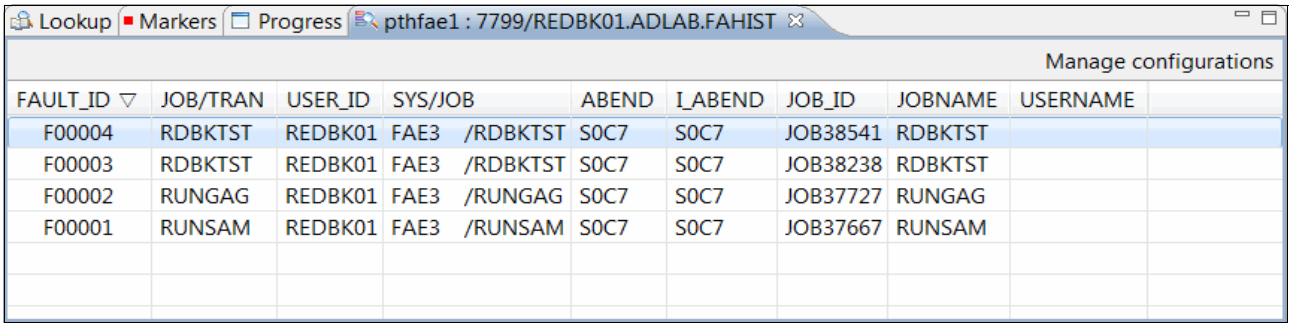
6.1.2 Starting Fault Analyzer for a batch program by using CICS Explorer

As an alternative to using Fault Analyzer under TSO/ISPF to view the details of the fault, we can view the same information through the Fault Analyzer perspective of CICS Explorer. For more information about these steps, see Chapter 2, “Introduction to the IBM Problem Determination Tools Plug-ins” on page 25.

Complete the following steps to start Fault Analyzer for a batch program by using CICS Explorer:

1. Start CICS Explorer and select the **Fault Analyzer** perspective. Right-click in the Fault Analyzer Objects view and add a history file that contains the fault entry.
2. Right-click the history file name in the Fault Analyzer objects view and select **Refresh** to ensure that the table contains the latest data from the history file.

   The table now contains the details of the fault that was written for our batch scenario (Fault_ID F00003), as shown in Figure 6-13.



FAULT_ID	JOB/TRAN	USER_ID	SYS/JOB	ABEND	I_ABEND	JOB_ID	JOBNAME	USERNAME
F00004	RDBKTST	REDBK01	FAE3 /RDBKTST	S0C7	S0C7	JOB38541	RDBKTST	
F00003	RDBKTST	REDBK01	FAE3 /RDBKTST	S0C7	S0C7	JOB38238	RDBKTST	
F00002	RUNGAG	REDBK01	FAE3 /RUNGAG	S0C7	S0C7	JOB37727	RUNGAG	
F00001	RUNSAM	REDBK01	FAE3 /RUNSAM	S0C7	S0C7	JOB37667	RUNSAM	

Figure 6-13 Updated table with our fault

Right-click the wanted Fault_ID, and select **Open**, to begin the analysis of this fault. The report for this fault is displayed in the main window of the Fault Analyzer perspective.

Initially, the Fault Summary and Synopsis are displayed, as shown in Figure 6-14. However, other Fault Analyzer reports, such as the Event Summary and the Abend Job Information, are available by selecting them from the tabs that are located below the main window.

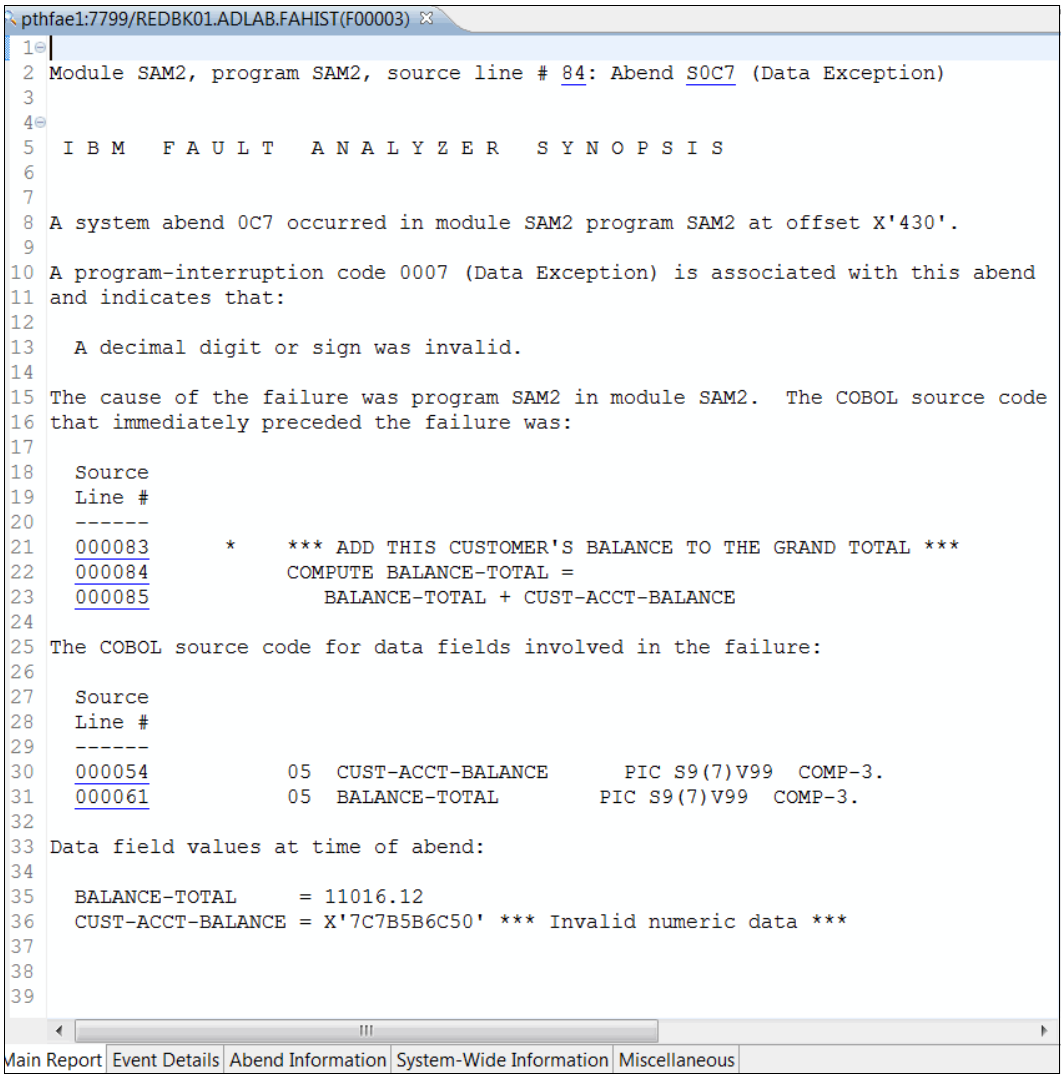


Figure 6-14 Fault Analyzer report

In this case, the synopsis reveals the type of abend (0C7), the offending lines of source code, and details of the data item that contained the invalid decimal data that led to the abend.

3. For more information, click the **Event Summary** tab to display the Event Summary Report for this fault. We see that there are three events reported on, the third event is the S0C7 abend.
4. Click this event to expand it, and various information that is related to the event is displayed. Scroll down to see the display of the data that is contained in the Linkage Section, as shown in Figure 6-15.

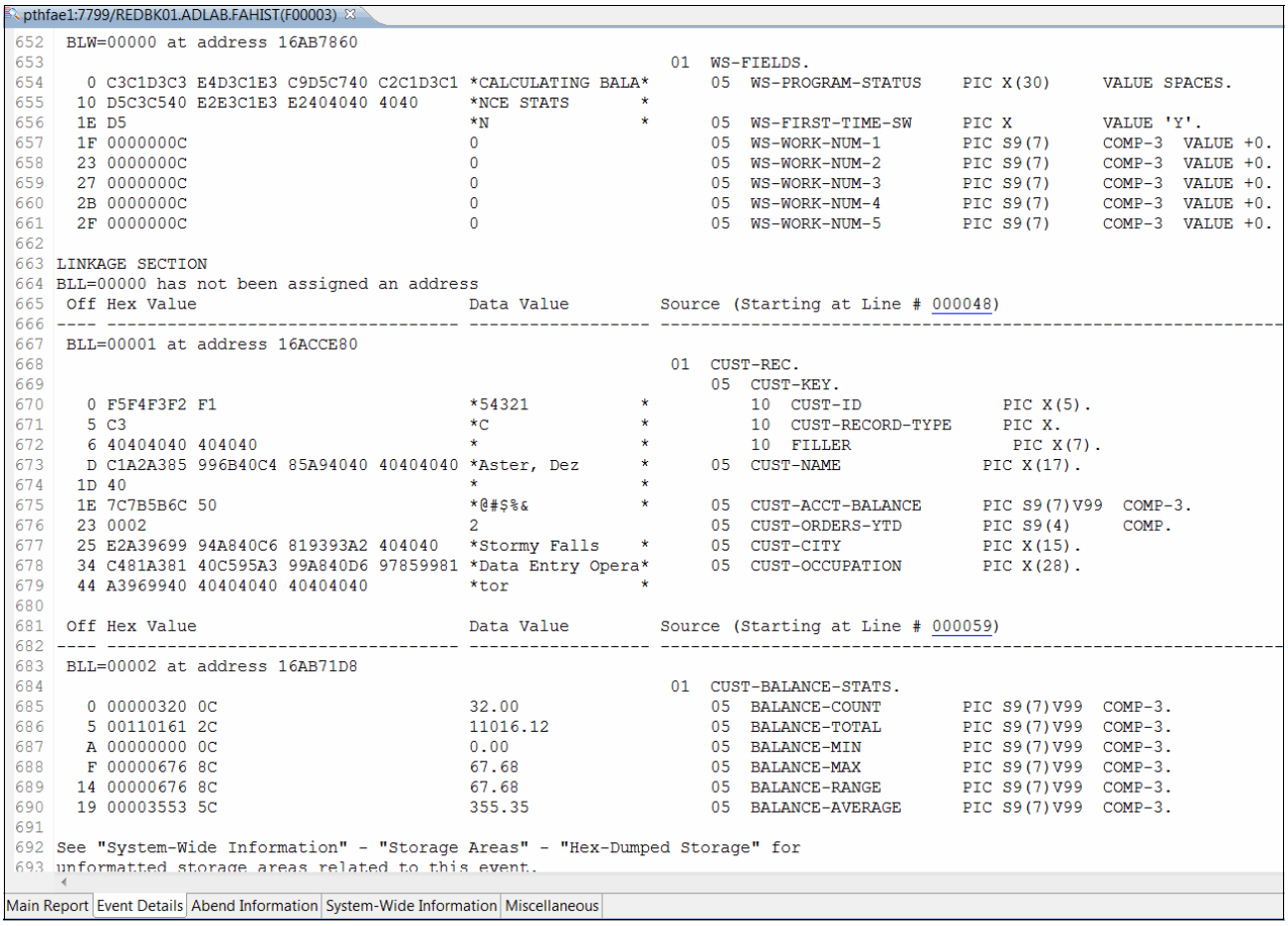


Figure 6-15 Linkage Section

The Linkage Section shows the data that is contained in the input record that was being processed at the time of the abend. We see that is a customer record for customer ID number 54321, for a customer with the name of Dez Aster. We can also see that the CUST-ACCT-BALANCE field in this record is contains invalid decimal data.

6.2 Using Debug Tool with a batch program

The next step in our scenario is to use Debug Tool to get a better understanding of what led to the abend in our batch job. We describe the use of Debug Tool from the Debug Tool Terminal Interface Manager, and from the Debug Tool perspective of CICS Explorer.

6.2.1 Starting Debug Tool for a batch program from a 3270 terminal

By using the Debug Tool Terminal Interface Manager, you can debug batch programs interactively by using a full-screen mode debugging session through an IBM VTAM terminal.

Assuming that the Debug Tool Terminal Interface Manager was installed and started, complete the following steps to start Debug tool for a batch program from a 3270 terminal:

1. Use a 3270 emulator session to connect to the Terminal Interface Manager. This session is separate from the session that you use to run the batch job that is to be debugged.
2. From this second terminal emulator session, enter your TSO user ID and password. This log in is not logging you on to TSO; it indicates to the Terminal Interface Manager that only your user ID is to be associated with this terminal logical units (LUs).

   You now see a panel that similar to the panel that is shown in Figure 6-16. This panel indicates that the terminal is ready to begin a debugging session for jobs that specify this user ID (TSS12) in their debug parameters.

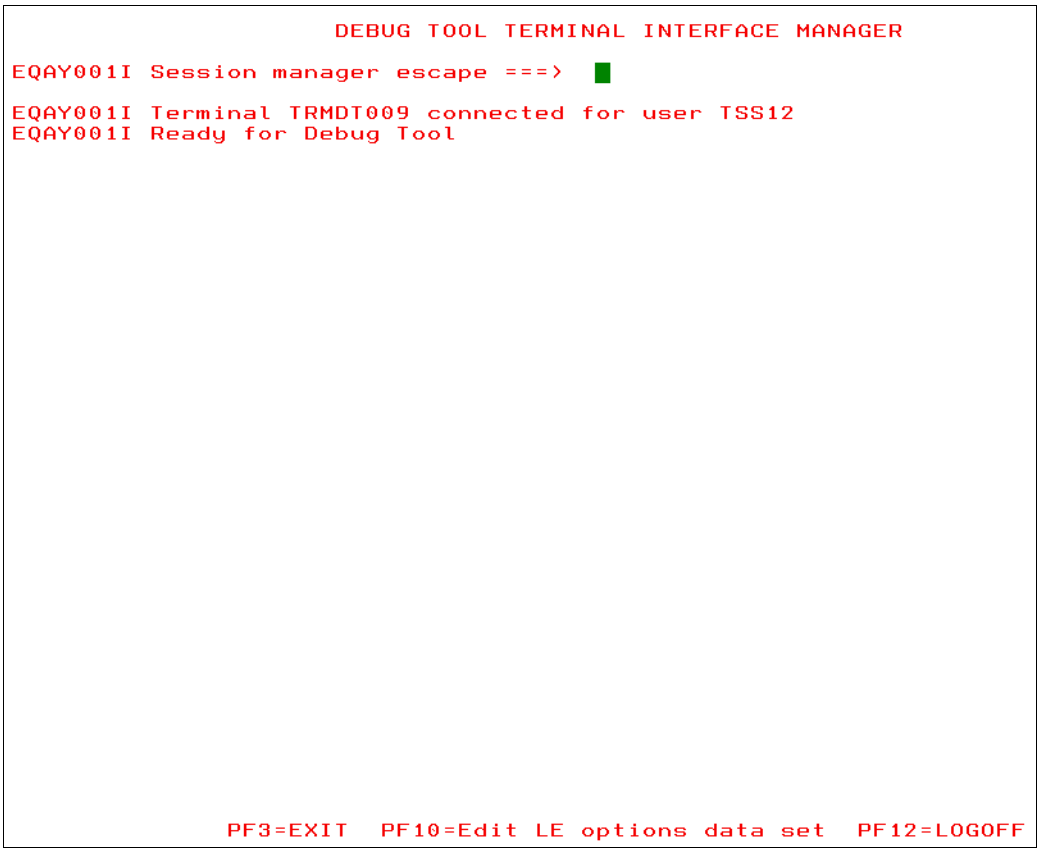


Figure 6-16 Terminal Interface Manager

You are now ready to submit the job from the original TSO session.

3. Pass the TEST parameter to the program in the JCL, as shown in Figure 6-17.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.ADLAB.JCL(XSAM) - 01.02                Columns 00001 00072
Command ===> sub                                        Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*      - - -  ADD A JOB CARD ABOVE THIS LINE  - - -
000004 //*
000005 //PRINT1  EXEC PGM=IDCAMS
000006 //SYSPRINT DD SYSOUT=*
000007 //FILE     DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000008 //SYSIN DD *
000009  PRINT INFILE(FILE) COUNT(1)
000010 //*
000011 //*
000012 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000013 //*************** DD'S FOR DEBUG TOOL ************************
000014 //CEEOPTS  DD *
000015 TEST(,,,VTAM%TSS12:)
000016 //**  //INSPLOG  DD SYSOUT=*
000017 //**  //EQADEBUG DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=SHR
000018 //**  //          DD DSN=&SYSUID..ADLAB.EQALANGX,DISP=SHR
000019 //**  //INSPPREF DD DSN=&SYSUID..ADLAB.DTPREF,DISP=SHR
000020 //*************************************************************
000021 //STEPLIB  DD DISP=SHR,DSN=&SYSUID..ADLAB.LOAD
000022 //**        DD DISP=SHR,DSN=DEBUG.V10R1.SEQAMOD   (UNCOMMENT IF NEEDED)
000023 //**        DD DISP=SHR,DSN=CEE.SCEERUN           (UNCOMMENT IF NEEDED)
000024 //CUSTFILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000025 //SYSPRINT DD SYSOUT=*
000026 //SYSOUT   DD SYSOUT=*
000027 //CUSTRPT  DD SYSOUT=*
000028 //CUSTOUT  DD SYSOUT=*
000029 //TRANFILE DD *
000030 *TRAN   (*  IN COL 1 IS A COMMENT)
000031 *-----
000032 PRINT      <== PRINT CUSTOMER LIST
000033 XXXX       BAD TRANSACTION
000034 TOTALS     <== PRINT TOTALS
000035 //*  ABEND      <== WILL CAUSE DIVIDE BY ZERO ABEND
```

Figure 6-17 Submit JCL with the CEEOPTS DD and TEST Parameter

The TEST parameter indicates that the program is to be processed by Debug Tool. The VTAM subparameter indicates that the debug session will run on a VTAM terminal. It is followed by the user ID (TSS12) to indicate to the listening Terminal Interface Manager that this program is waiting to debug.

When the job is submitted and the program is started, the Debug Tool Terminal Interface Manager session becomes active, as shown in Figure 6-18.

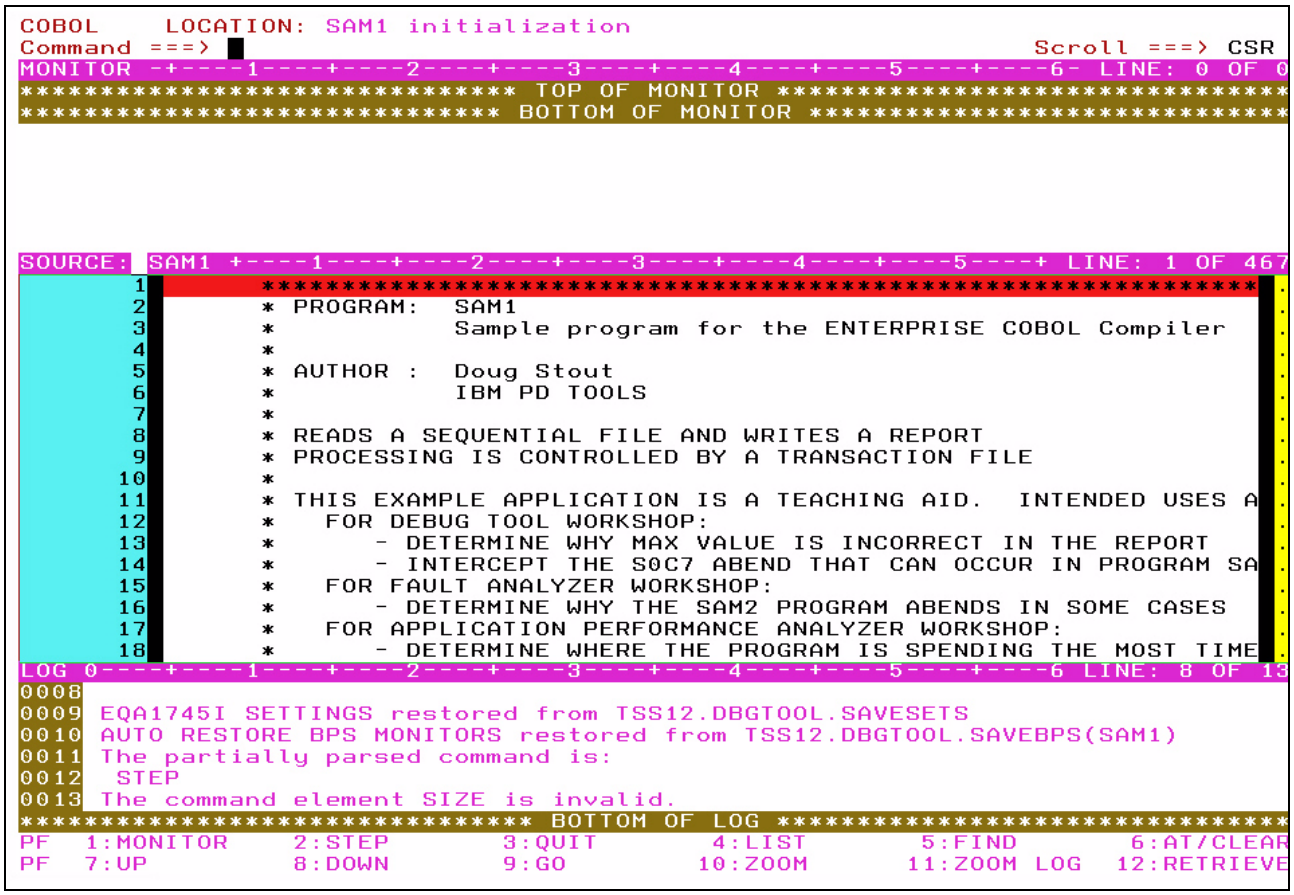


Figure 6-18 The Terminal Interface Manager is now active

A monitor window, a source window, and a log window are shown. We can display more of the information that we are interested in by zooming in on a particular window by using the Zoom command (PF 10).

4. With the cursor on the command line or in the source window, press PF10 to display the panel that is shown in Figure 6-19.

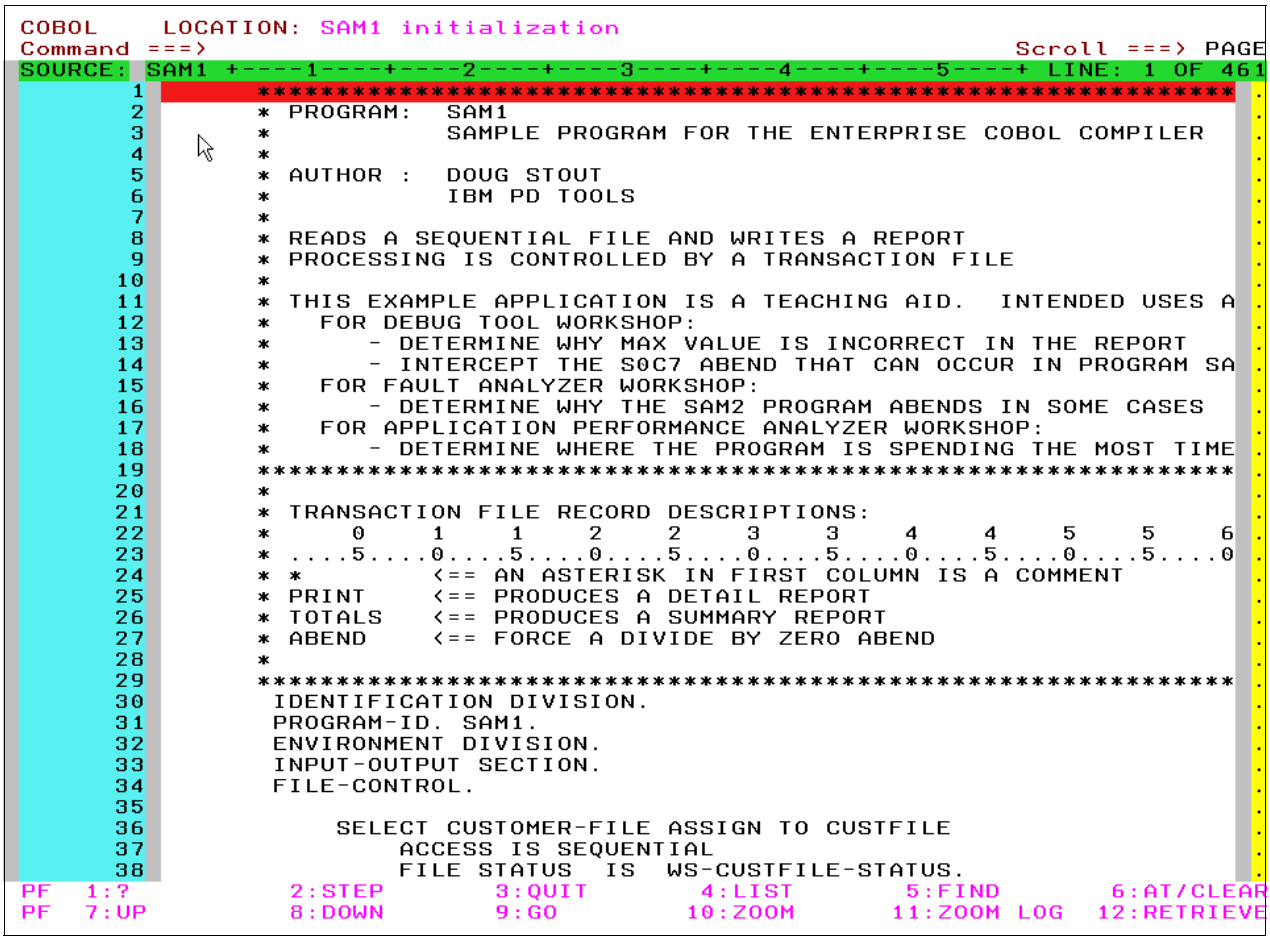


Figure 6-19 Enter PF10 to zoom

From here, you can start any of the Debug Tool functions, such as stepping or running through the program, setting breakpoints, and monitoring variables.

5. Press PF9 without setting any breakpoints, which allows the program to run to completion.

As shown in Figure 6-20, the program runs until it reaches the line of code that causes the abend that we were expecting.

```
COBOL     LOCATION:  SAM2  :> 84.1
Command ===>                                                 Scroll ===> PAGE
SOURCE:  SAM2 +----1----+----2----+----3----+----4----+----5---- LINE:  76 OF 113
        76              MOVE 'PROGRAM ENDED' TO WS-PROGRAM-STATUS.
        77              GOBACK.
        78
        79           100-CALC-BALANCE-STATISTICS.
        80              MOVE  'CALCULATING BALANCE STATS' TO WS-PROGRAM-STATUS.
        81         *    *** INCREMENT RECORD COUNT ***
        82              ADD +1 TO BALANCE-COUNT
        83         *    *** ADD THIS CUSTOMER'S BALANCE TO THE GRAND TOTAL ***
        84              COMPUTE BALANCE-TOTAL =
        85                 BALANCE-TOTAL + CUST-ACCT-BALANCE
        86         *    *** CALCULATE AVERAGE ***
        87              COMPUTE BALANCE-AVERAGE =
        88                 BALANCE-TOTAL / BALANCE-COUNT
        89         *    *** CALCULATE MINIMUM ***
        90              IF WS-FIRST-TIME-SW = 'Y'
        91                 MOVE CUST-ACCT-BALANCE TO BALANCE-MIN.
        92              IF CUST-ACCT-BALANCE < BALANCE-MIN
        93                 MOVE CUST-ACCT-BALANCE TO BALANCE-MIN.
        94         *    *** CALCULATE MAXIMUM ***
        95         *    ***    THERE IS A BUG CALCULATING THE MAXIMUM.   ***
        96         *    ***              CAN YOU FIND IT?                ***
        97              IF WS-FIRST-TIME-SW = 'Y'
        98                 MOVE CUST-ACCT-BALANCE TO BALANCE-MAX
        99              IF CUST-ACCT-BALANCE > BALANCE-MAX
       100                 MOVE CUST-ACCT-BALANCE TO BALANCE-MAX.
       101         *    *** CALCULATE RANGE ***
       102              COMPUTE BALANCE-RANGE = BALANCE-MAX - BALANCE-MIN.
       103
       104           500-INIT-STATISTICS.
       105              MOVE 'ZEROING STATS VARIABLES' TO WS-PROGRAM-STATUS.
       106              MOVE 0   TO BALANCE-COUNT.
       107              MOVE 0   TO BALANCE-TOTAL.
       108              MOVE 0   TO BALANCE-MIN.
       109              MOVE 0   TO BALANCE-MAX.
       110              MOVE 0   TO BALANCE-RANGE.
       111              MOVE 0   TO BALANCE-AVERAGE.
       112
       113         *  END OF PROGRAM SAM2
PF   1:?          2:STEP        3:QUIT        4:LIST        5:FIND        6:AT/CLEAR
PF   7:UP         8:DOWN        9:GO         10:ZOOM       11:ZOOM LOG   12:RETRIEVE
```

Figure 6-20 Line 84 is the line that causes the abend

6. Press the Zoom key (PF10) to see the log messages that refer to this interruption, as shown in Figure 6-21.

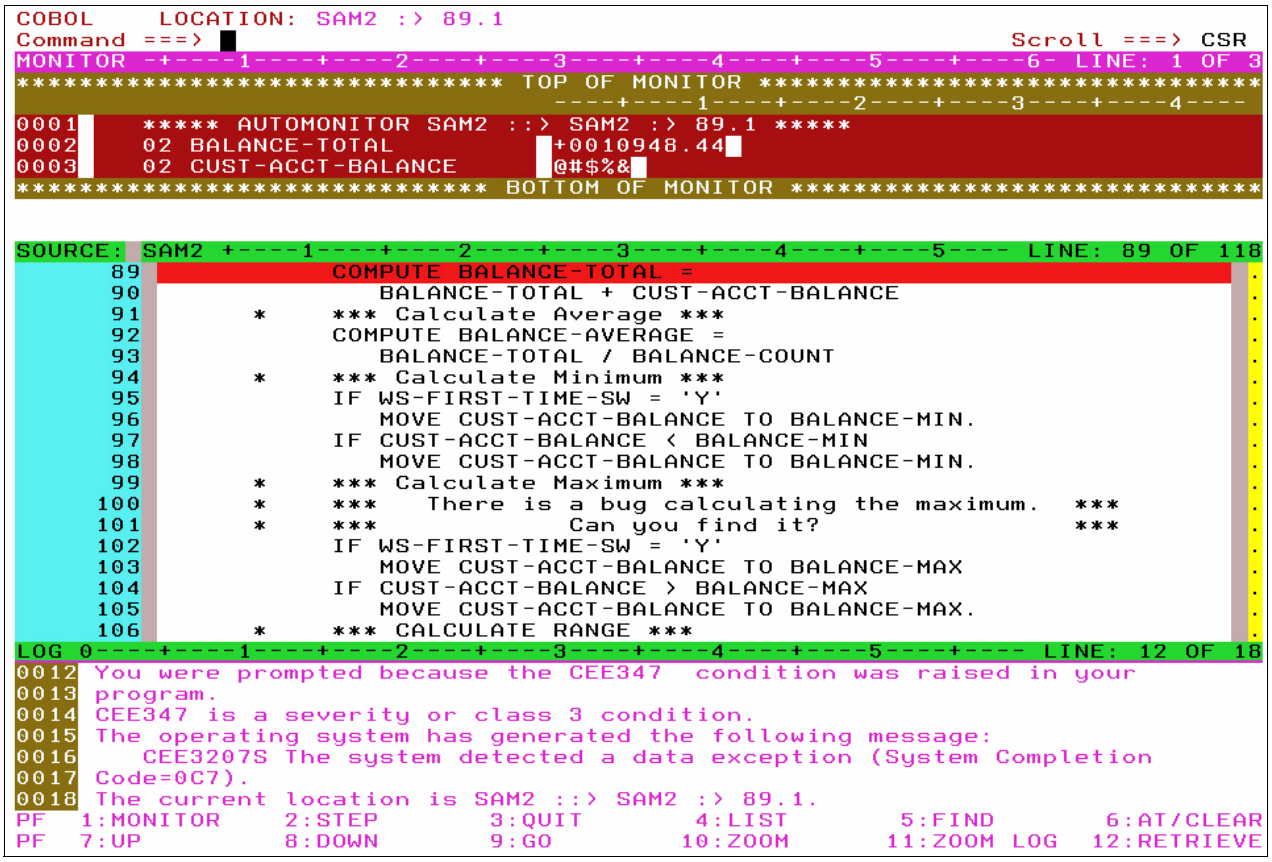


Figure 6-21 Zoom

The messages confirm that the abend is caused by a data exception. The offending line of code adds two decimal values together, so we suspect that one of the two variables contains the invalid decimal data.

The Monitor window shows the variables and their values, which are based on the current position in the source (line 89). The SET AUTO ON command is in effect.

In Figure 6-22, we see that the CUST-ACCT-BALANCE variable contains the invalid data.

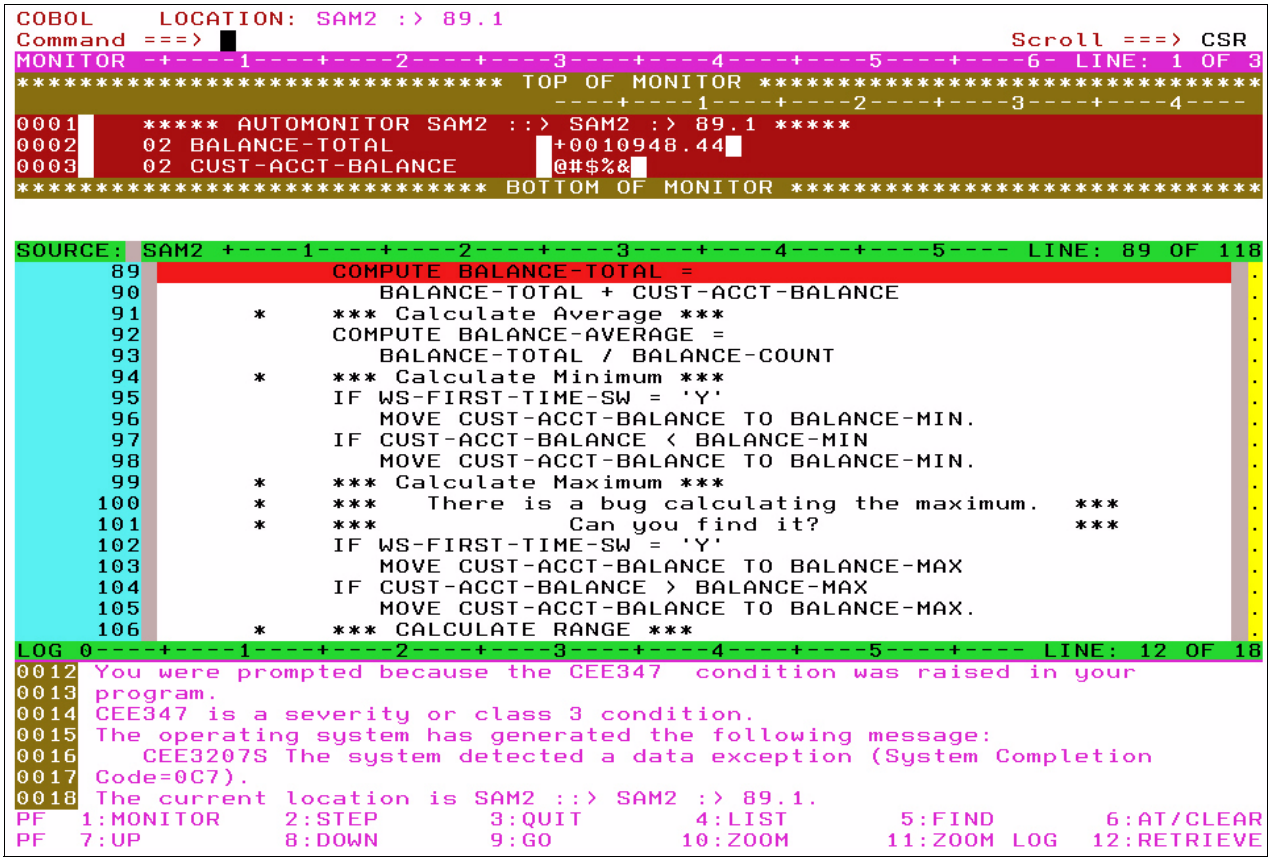


Figure 6-22 CUST-ACCT-BALANCE variable

Next, we might want to obtain more information about the actual record that contains the invalid data. The data structure that maps the customer input records in our COBOL program is CUST-REC.

7. Issue the **LIST CUST-REC** command. The result is shown in Figure 6-23.

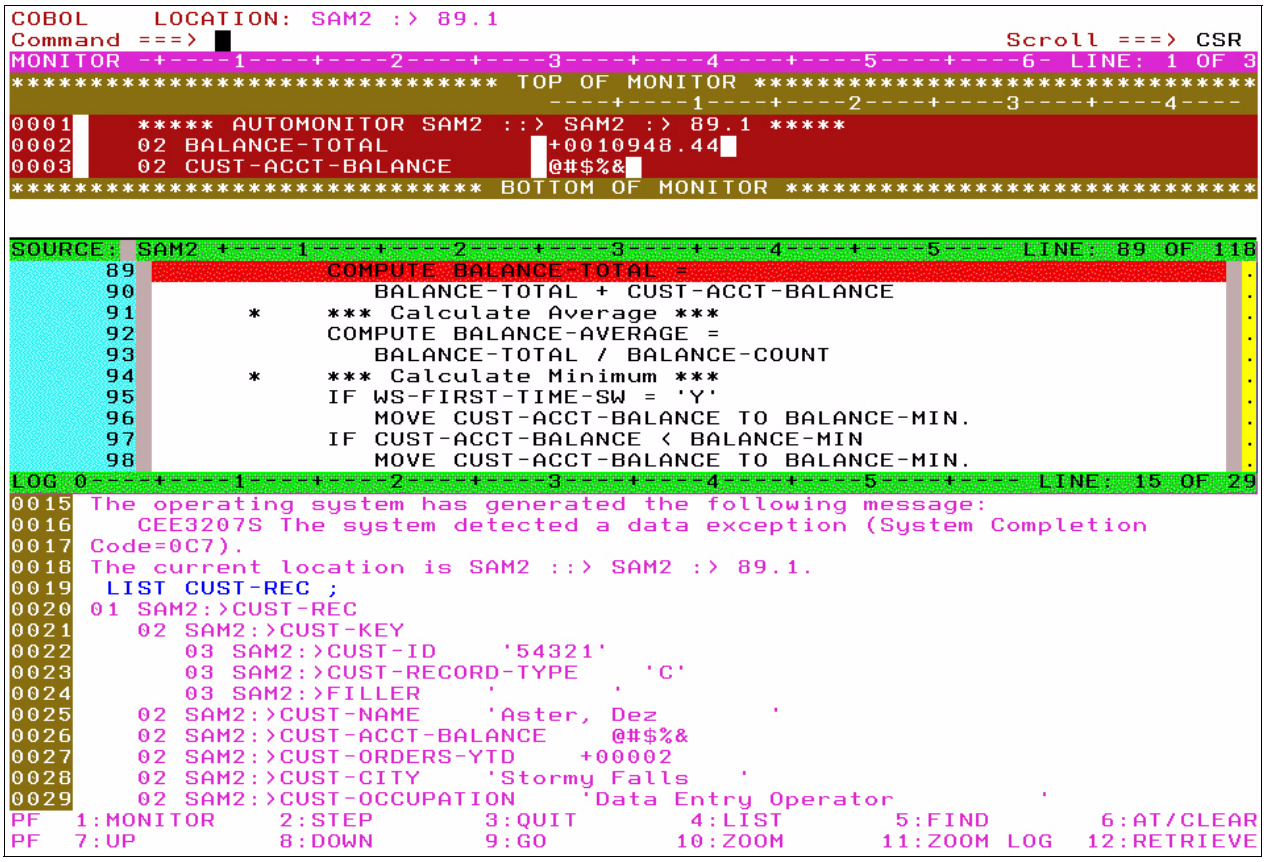


Figure 6-23 Results of the LIST CUST-REC

We can see that the offending customer input record is for customer ID number 54321, for a customer with the name of Dez Aster.

6.2.2 Starting Debug Tool for a batch program from CICS Explorer

Complete the following steps to establish a debug session in CICS Explorer for a batch program:

1. Start CICS Explorer and select the **Debug** perspective, as shown in Figure 6-24.

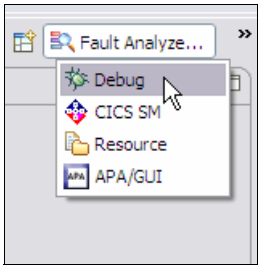


Figure 6-24 Debug perspective

We must determine the IP address of our workstation because this information is required in the TEST parameter of our batch program. The IP address is used to establish the connection to this debug session.

2. From the Debug perspective, click the **Debug** UI daemon icon and select **Get Workstation IP** from the pull-down menu, as shown in Figure 6-25.

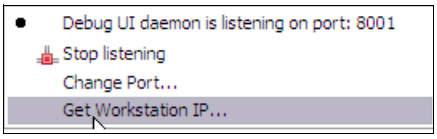


Figure 6-25 Get Workstation IP

The workstation IP address is displayed (as shown in Figure 6-26) so we make a note of it.

Figure 6-26 Workstation IP displayed

Now we are ready to submit the job from our TSO session.

3. Indicate that the program is to be processed by Debug Tool by passing the TEST parameter to the program in the JCL, as shown in Figure 6-27.

   We use the TCP/IP subparameter to indicate that the debug session will run on a distributed workstation. It is followed by the IP address that we obtained from CICS Explorer, which specifies the connection to the listening debug session.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Iest  Help
EDIT       TSS12.ADLAB.JCL(XSAM) - 01.03                Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*      - - -  ADD A JOB CARD ABOVE THIS LINE  - - -
000004 //*
000005 //PRINT1  EXEC PGM=IDCAMS
000006 //SYSPRINT DD SYSOUT=*
000007 //FILE     DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000008 //SYSIN DD *
000009  PRINT INFILE(FILE) COUNT(1)
000010 //*
000011 //*
000012 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000013 //*************** DD'S FOR DEBUG TOOL ************************
000014 //CEEOPTS  DD *
000015 TEST(,,,TCPIP&9.48.124.11%8001:)
000016 //**  //INSPLOG  DD SYSOUT=*
000017 //**  //EQADEBUG DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=SHR
000018 //**  //          DD DSN=&SYSUID..ADLAB.EQALANGX,DISP=SHR
000019 //**  //INSPPREF DD DSN=&SYSUID..ADLAB.DTPREF,DISP=SHR
000020 //*************************************************************
000021 //STEPLIB  DD DISP=SHR,DSN=&SYSUID..ADLAB.LOAD
000022 //**        DD DISP=SHR,DSN=DEBUG.V10R1.SEQAMOD   (UNCOMMENT IF NEEDED)
000023 //**        DD DISP=SHR,DSN=CEE.SCEERUN           (UNCOMMENT IF NEEDED)
000024 //CUSTFILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000025 //SYSPRINT DD SYSOUT=*
000026 //SYSOUT   DD SYSOUT=*
000027 //CUSTRPT  DD SYSOUT=*
000028 //CUSTOUT  DD SYSOUT=*
000029 //TRANFILE DD *
000030 *TRAN   (*  IN COL 1 IS A COMMENT)
000031 *-----
000032 PRINT      <== PRINT CUSTOMER LIST
000033 XXXX       BAD TRANSACTION
000034 TOTALS     <== PRINT TOTALS
000035 //*  ABEND      <== WILL CAUSE DIVIDE BY ZERO ABEND
```

Figure 6-27 JCL for Test parameter

4. Submit the job. When the program is started, the Debug perspective in CICS Explorer becomes active, as shown in Figure 6-28.

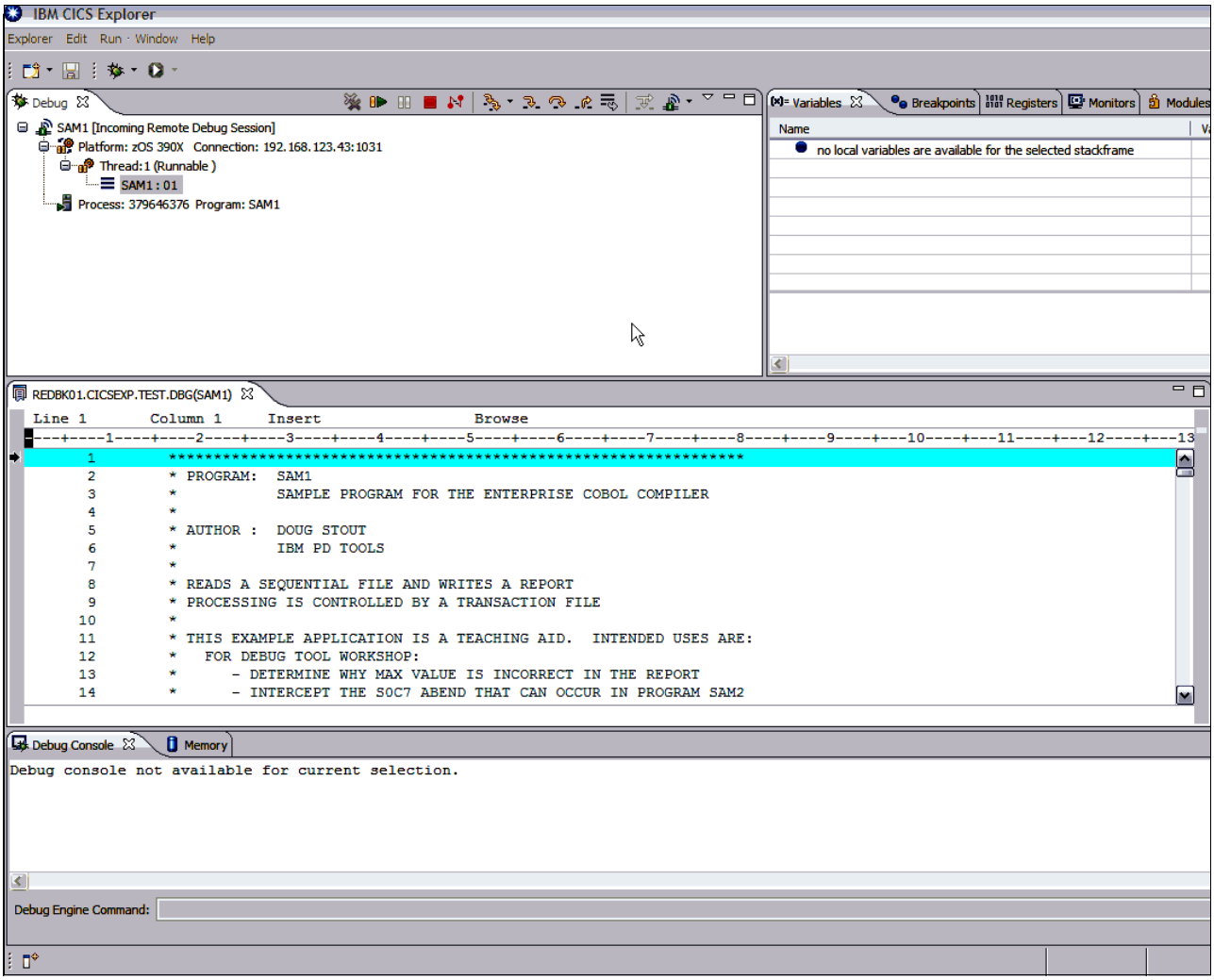


Figure 6-28 Debug perspective in CICS Explorer is active

The same Debug Tool functions and commands are available as described in section 6.2, “Using Debug Tool with a batch program” on page 220. The Variables view displays the output from Automonitor Current by default, as shown in Figure 6-29.

Monitor CUST-REC by double-clicking CUST-REC in the source window and right-click to choose **Monitor Expression**.

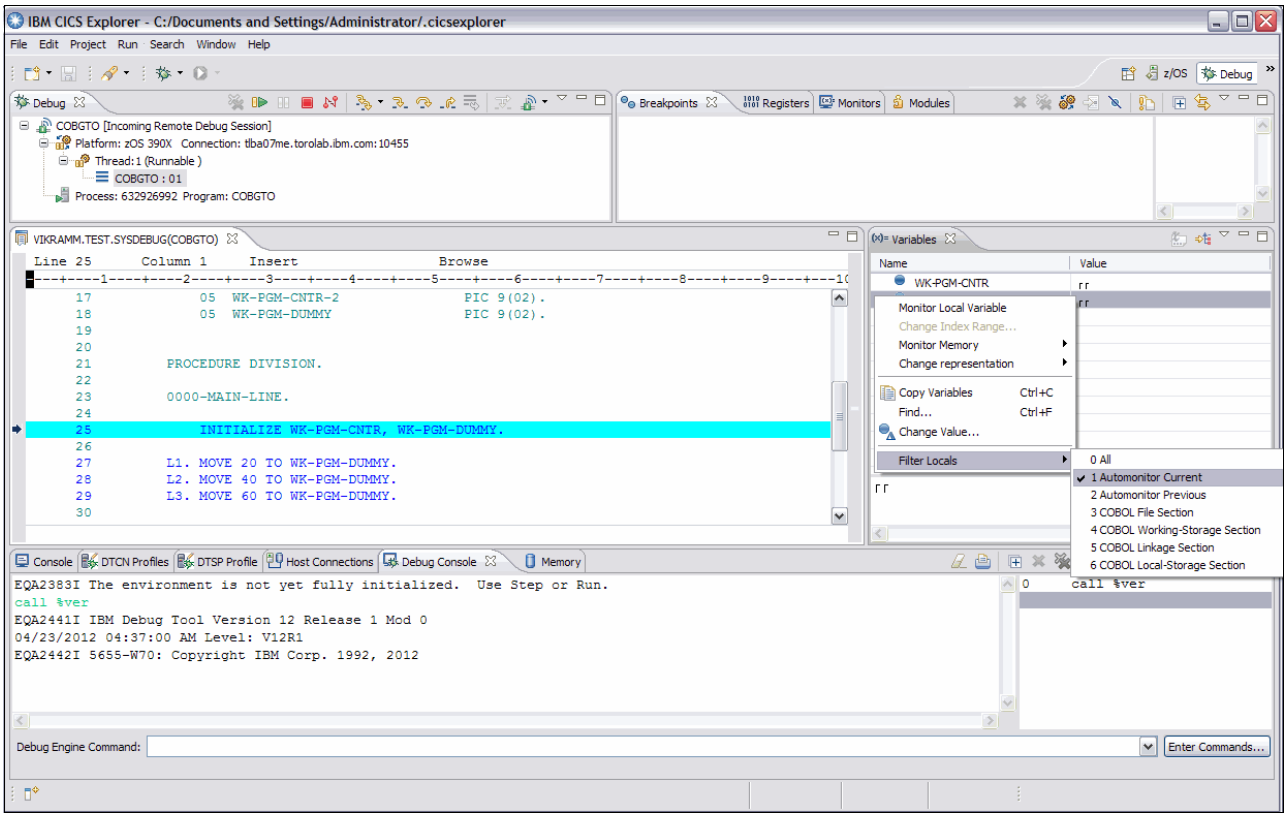


Figure 6-29 Right-click inside the Variables view and select Filter Locals to see a list of choices

6.3 Using File Manager to fix the problem

Next, we complete the following steps to use File Manager attempt to amend the invalid data in the input file:

1. Start File Manager from an ISPF session (typically by entering FM from an ISPF session).
2. From the File Manager Primary Option menu, select **option 2**, as shown in Figure 6-30.

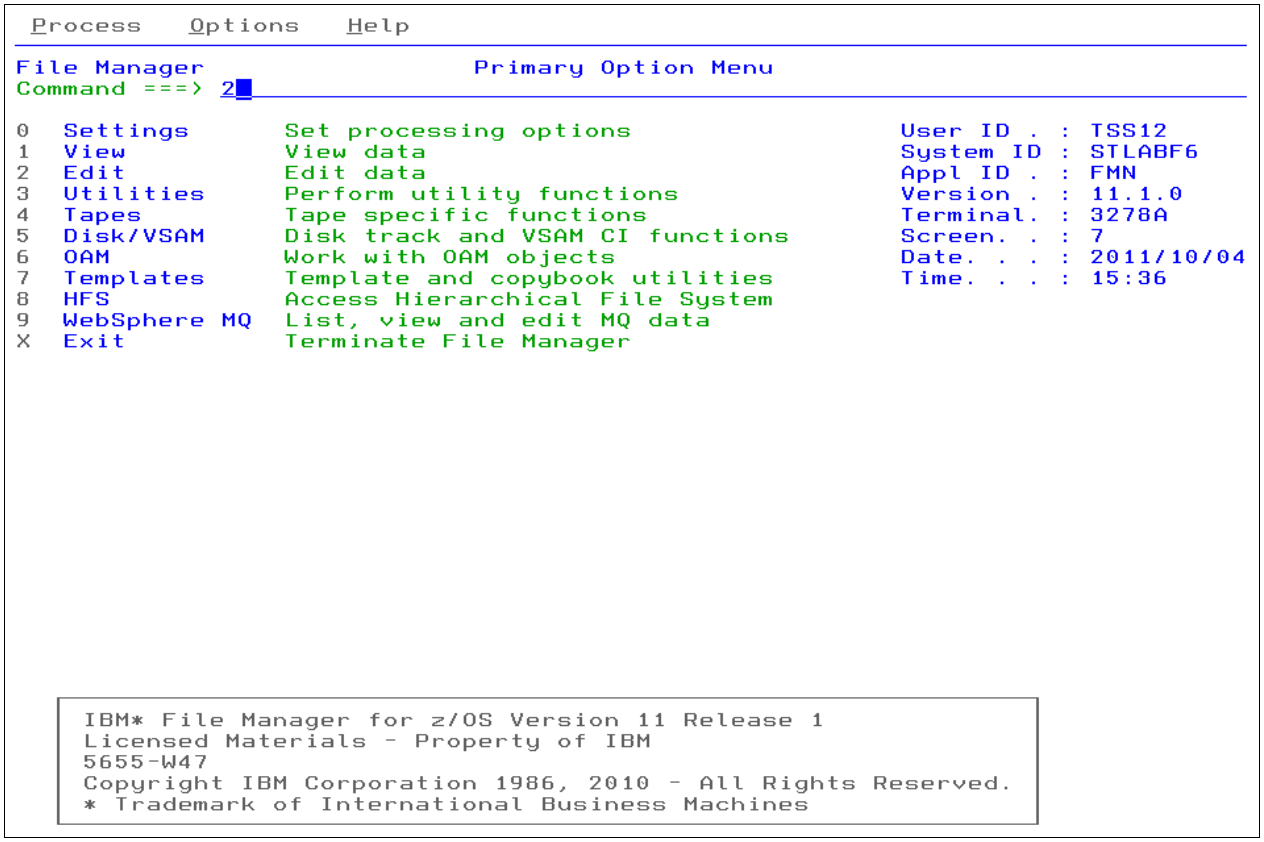


Figure 6-30 FM primary menu

3. In the Edit Entry panel (see Figure 6-31), specify the name of the input data set. In this case, the record format of the input file is mapped by a copybook file, which we also specify in this panel.

```
 Process   Options   Help
File Manager                 Edit Entry Panel
Command ===> 

Input Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name ADLAB.FILES(CUST2FA)                              +
   Member . . . . . .               (Blank or pattern for member list)
   Volume serial  . .               (If not cataloged)
   Start position . .                                +
   Record limit . . .               Record sampling _
   Inplace edit . . . _             (Prevent inserts and deletes)
Copybook or Template:
   Data set name  . . ADLAB.TEMPLATE(CUST2)
   Member . . . . . .               (Blank or pattern for member list)
Processing Options:
 Copybook/template    Start position type   Enter "/" to select option
 1   1. Above         _    1. Key           _   Edit template   _ Type (1,2,S)
     2. Previous           2. RBA           _   Include only selected records
     3. None               3. Record number _   Binary mode, reclen 80
     4. Create dynamic     4. Formatted key _   Create audit trail
```

Figure 6-31 Edit Entry panel

From our earlier fault analysis and debugging, we know that the invalid data is in a customer record for a customer with an ID number of 54321 and the name Dez Aster.

4. Use the Find Error (FE) command with the ALL parameter, as shown in Figure 6-32.

```
 Process   Options   Help
Edit             TSS12.ADLAB.FILES(CUST2FA)                        Top of 61
Command ===> fe all                                                 Scroll CSR
                                                   Record AT TOP    Format TABL
       CUST-ID RECORD-TYPE FILLER NAME             ACCT-BALANCE ORDERS-YTD
       #3      #4          #5     #6                         #7         #8
       AN 1:5  AN 6:1      AN 7:7 AN 14:17             PD 31:5    BI 36:2
       <--->   -           <---+-> <---+----1----+->  <---+----1>    <---+>
****** ****  Top of data  ****
000001 01001   C                  Lynn, Amanda           67.68          9
000002 02200   C                  Graham, Anna          610.05         10
000003 02202   C                  Major, Art           1234.56          5
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000005 03003   C                  Prentice, Anna        396.63          7
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 2 Line(s) suppressed
000008 03390   C                  Deeds, Darren          74.00          3
000009 05500   C                  Parker, Ford          233.27          5
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000011 06101   C                  Early, Brighton       311.08         10
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000013 06106   C                  Lander, Annette       489.84          7
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000015 06711   C                  Dubree, Dustin        192.98          1
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 4 Line(s) suppressed
000020 06900   C                  Bacon, Chris P.      1001.01          0
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 2 Line(s) suppressed
000023 07008   C                  Houston, Roger        296.97         10
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000025 07044   C                  Schauer, April         88.83          7
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 2 Line(s) suppressed
000028 07077   C                  Mann, Mr. E.          621.05          1
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 1 Line(s) suppressed
000030 07707   C                  Clime, Hilda          232.20          9
- - -  - - - - - - PRODUCT-RECORD - - - - - - - - - 3 Line(s) suppressed
000034 07870   C                  Furst, April          122.15          1
000035 07880   C                  Marx, Audrey          450.51          2
000036 08085   C                  Moore, Adeline        498.95          3
000037 10207   C                  Loblaw, Bob            11.11          0
000038 10290   C                  Well, Alice           731.26          4
```

Figure 6-32 Entering the FE command

The FE All command moved to the Bad Record and gave us a count, as shown in Figure 6-33.

```
 Process    Options    Help
Edit              TSS12.ADLAB.FILES(CUST2FA)                    1 error(s) found
Command ===> ____________________________________________________ Scroll CSR
                                                 Record 60       Format TABL
        CUST-ID RECORD-TYPE FILLER  NAME              ACCT-BALANCE ORDERS-YTD
        #3      #4          #5      #6                      #7          #8
        AN 1:5  AN 6:1      AN 7:7  AN 14:17              PD 31:5     BI 36:2
        <--->   -           <---+-> <---+----1----+->   <---+----1>    <---+>
000060  54321   C                   Aster, Dez           **********        2
000061  55555   C                   DeMann, Hugh            1234.00        3
******  ****  End of data  ****
```

Figure 6-33 Results of FE command

5. As another way to find the bad data, use the Find command (by entering **f 'Dez Aster'** on the command line).
6. Run the **HEX ON** command and type over the bad data or use the **CE** command to change the bad data value, as shown in Figure 6-34.

```
 Process    Options    Help
Edit              TSS12.ADLAB.FILES(CUST2FA)                    1 error(s) found
Command ===> hex on; ce 1234 ____________________________________ Scroll CSR
                                                 Record 60       Format TABL
        CUST-ID RECORD-TYPE FILLER  NAME              ACCT-BALANCE ORDERS-YTD
        #3      #4          #5      #6                      #7          #8
        AN 1:5  AN 6:1      AN 7:7  AN 14:17              PD 31:5     BI 36:2
        <--->   -           <---+-> <---+----1----+->   <---+----1>    <---+>
000060  54321   C                   Aster, Dez           **********        2
000061  55555   C                   DeMann, Hugh            1234.00        3
******  ****  End of data  ****
```

Figure 6-34 CE command to change data

Here we ran the **HEX ON** command and changed the bad data to the value 1234, as shown in Figure 6-35.

```
 Process    Options    Help
Edit              TSS12.ADLAB.FILES(CUST2FA)                  1 error(s) changed
Command ===> ____________________________________________________ Scroll CSR
                                                 Record 60       Format TABL
        CUST-ID RECORD-TYPE FILLER  NAME              ACCT-BALANCE ORDERS-YTD
        #3      #4          #5      #6                      #7          #8
        AN 1:5  AN 6:1      AN 7:7  AN 14:17              PD 31:5     BI 36:2
        <--->   -           <---+-> <---+----1----+->   <---+----1>    <---+>
000060  54321   C                   Aster, Dez              1234.00        2
        FFFFF   C           4444444 CAA8964C8A4444444         00240       00
        54321   3           0000000 12359B04590000000         0130C       02

000061  55555   C                   DeMann, Hugh            1234.00        3
        FFFFF   C           4444444 C8D89964CA8844444         00240       00
        55555   3           0000000 454155B0847800000         0130C       03

******  ****  End of data  ****
```

Figure 6-35 HEX on

7. With the invalid decimal data field amended, resubmit the job and it runs successfully to completion.

8. Optionally, use File Manager in batch to fix the bad data, as shown in Figure 6-36.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FIXBADNM) - 01.00        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000010 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000020 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000100 //*
000200 //STEP001 EXEC PGM=FMNMAIN
000300 //SYSOUT    DD  SYSOUT=*
000400 //SYSPRINT  DD  SYSOUT=*
000500 //FMNTSPRT  DD  SYSOUT=*
000600 //FMNSRTP   DD  SYSOUT=*
000700 //SYSTERM   DD  SYSOUT=*
000800 //DDDUMMY   DD  DUMMY
000900 //DD01      DD  DISP=SHR,DSN=&SYSUID..ADLAB.FILES(CUST2FA)
001200 //SYSIN     DD  *
001300 $$FILEM DSU  INPUT=DD01,
001500 $$FILEM       PROC=*
001600  IF FLD_TYPE(31,5,P) = 0
001700  THEN DO
001800   OVLY_OUT('123400',31,5,P)
001810   RETURN                              /* PROCESS RECORD                */
001900  END
002000  RETURN DROP                          /* NO CRITERIA MATCH, DROP       */
002100 /+
****** ***************************** Bottom of Data *****************************
```

Figure 6-36 Fixing the bad data from a batch job

6.4 Using Application Performance Analyzer with a batch program

The test program that we use in this scenario consists of a COBOL program, which is running as part of a batch job. The main module of the program, SAM1, processes two input files and produces a report. It calls two other programs, SAM2 and SAM3, as part of this process. This is the same program that was used in the previous Fault Analyzer and Debug Tool scenarios. However, for this scenario, the invalid input data was amended by using File Manager, so we expect the program to run to completion without abending.

6.4.1 Starting APA for a batch program from an ISPF session

For this scenario, before we submit the batch job, we must create an observation request to identify to APA the batch jobs that we are interested in analyzing.

Complete the following steps to start APA for a batch program from an ISPF session:

1. Start APA from TSO/ISPF and enter NEW as a primary command, or type over an existing request number and pressing Enter.
2. Create an observation from the command line, as shown in Figure 6-37.

```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZA)                Row 00001 of 00104
Command ===> new                                              Scroll ===> CSR

 NEW      To define a new measurement
 TNEW     To define a threshold measurement
 CONNECT  To connect to another instance of the measurement task
 VERSION  To display version information for all instances
 IMPORT   To IMPORT a previously Exported sample file
 HIDE     To remove these commands from the display (recommended)
 /        On top of any ReqNum to get a list of the line commands

ReqNum   Owned By   Description      Job Name   Date/Time        Samples   Status

 9955     #845409   P33913.WORKRND   T263RAFT   Mar-6  15:28      2,603   Cancel
 9954     #845409   P33913.NODT      T263RAFT   Mar-6  15:28      1,000   Ended
 9953     #845409   P33913.NORMAL    T263RAFT   Mar-6  15:27      4,134   Cancel
 9730     MACHIN2   v9ref-uc17       CICSC32F   Feb-28  7:04     99,999   Ended
 9729     MACHIN2   v9-uc3           MQPUT      Feb-28  7:04        774   Ended
 9728     MACHIN2   v9               MQPUT      Feb-28  7:02        774   Ended
 9660     MACHIN2   v10L-uc21x15     CICSC41F   Feb-22 10:49     77,777   REPEAT
 9636     MACHIN2   v10K-uc30        -          Jan-13 11:05     22,222   Sched
 9635     MACHIN2   v10Kref-uc21x5   CICSC41F   Jan-13 11:03     33,333   REPEAT
 9573     MACHIN2   v9ref-uc17       CICSC32F   Jan-10  7:29     99,999   Ended
 9572     MACHIN2   v9-uc3           MQPUT      Jan-10  7:29        774   Ended
 9571     MACHIN2   del              MQPUT      Jan-10  7:27        774   Ended
 9570     MACHIN2   v9               MQPUT      Jan-10  7:27        774   Ended
 9334     MACHIN2   v10K-uc21x15     CICSC32F   Jan-5   9:06     55,555   REPEAT
 9332     CHIDGEY                    CHIDGEYS   Dec-29  9:24     10,000   REPEAT
 9284     TSS16     Dougs SAM2V IO   TSS16V     Dec-9   9:49      8,048   Ended
 9083     MACHIN2   v9ref-uc17       CICSC32F   Nov-1   9:41     99,999   Ended
 9082     MACHIN2   v9-uc3           MQPUT      Nov-1   9:41        774   Ended
 9081     MACHIN2   v9               MQPUT      Nov-1   9:33        774   Ended
 8928     MACHIN2   v9ref-uc17       CICSC32F   Sep-30  6:49     99,999   Ended
 8926     MACHIN2   v9ref-uc17       CICSC32F   Sep-20 10:21     99,999   Ended
 8925     MACHIN2   v9-uc3           MQPUT      Sep-20 10:21        774   Ended
 8924     D2BAMJT   Ford IMS         SDIMP74K   Sep-17  7:22        407   Ended
 8913     MACHIN2   v9ref-uc17       CICSC32F   Sep-15  7:09     99,999   Ended
 8912     MACHIN2   v9-uc3           MQPUT      Sep-15  7:09        774   Ended
```

Figure 6-37 APA new observation

The Schedule New Measurement panel is displayed, as shown in Figure 6-38.

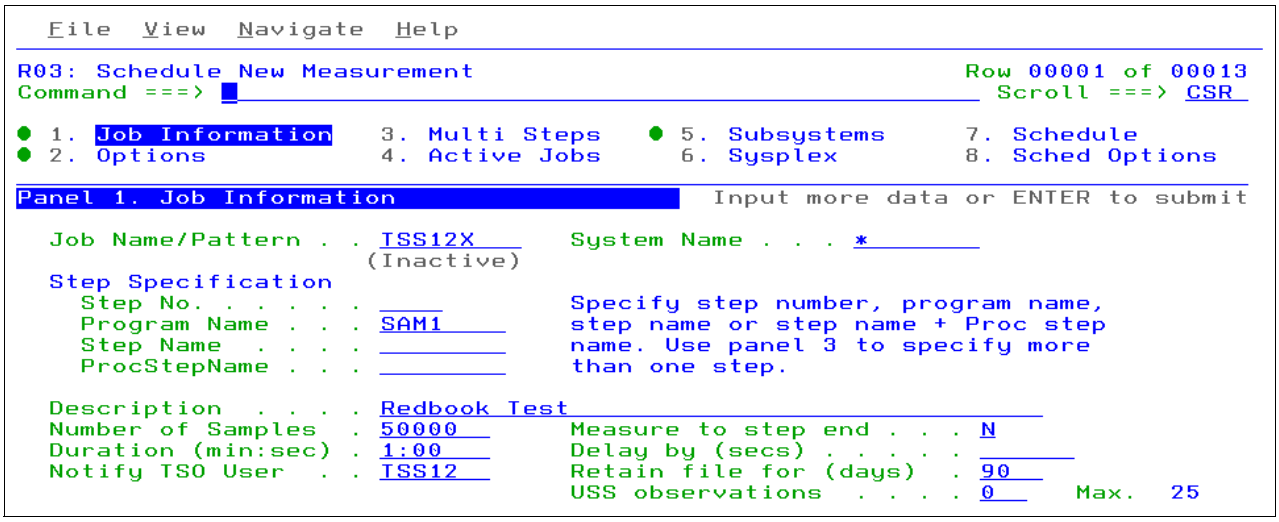


Figure 6-38 Schedule a new measurement

3. Identify the job to be analyzed by specifying the name of the job and the specific program or job step. Also, provide a description for this observation and specify the number of samples that APA must attempt to take when the program is run.
4. Submit the job.

   As is shown in Figure 6-39, no other parameters are required for APA to be started if the job name and program or job step name matches the information that was specified when we created the observation.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Iest  Help
EDIT       TSS12.ADLAB.JCL(XSAM) - 01.04                Columns 00001 00072
Command ===> sub                                         Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*      - - -  ADD A JOB CARD ABOVE THIS LINE  - - -
000004 //*
000005 //PRINT1  EXEC PGM=IDCAMS
000006 //SYSPRINT DD SYSOUT=*
000007 //FILE     DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000008 //SYSIN DD *
000009  PRINT INFILE(FILE) COUNT(1)
000010 //*
000011 //*
000012 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000013 //*************** DD'S FOR DEBUG TOOL ************************
000014 //**  //CEEOPTS  DD *
000015 //**  TEST(,,,TCPIP&9.48.124.11%8001:)
000016 //**  //INSPLOG  DD SYSOUT=*
000017 //**  //EQADEBUG DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=SHR
000018 //**  //          DD DSN=&SYSUID..ADLAB.EQALANGX,DISP=SHR
000019 //**  //INSPPREF DD DSN=&SYSUID..ADLAB.DTPREF,DISP=SHR
000020 //*************************************************************
000021 //STEPLIB  DD DISP=SHR,DSN=&SYSUID..ADLAB.LOAD
000022 //**        DD DISP=SHR,DSN=DEBUG.V10R1.SEQAMOD   (UNCOMMENT IF NEEDED)
000023 //**        DD DISP=SHR,DSN=CEE.SCEERUN           (UNCOMMENT IF NEEDED)
000024 //CUSTFILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000025 //SYSPRINT DD SYSOUT=*
000026 //SYSOUT   DD SYSOUT=*
000027 //CUSTRPT  DD SYSOUT=*
000028 //CUSTOUT  DD SYSOUT=*
000029 //TRANFILE DD *
000030 *TRAN   (*  IN COL 1 IS A COMMENT)
000031 *-----
000032 PRINT      <== PRINT CUSTOMER LIST
000033 XXXX       BAD TRANSACTION
000034 TOTALS     <== PRINT TOTALS
000035 //*  ABEND      <== WILL CAUSE DIVIDE BY ZERO ABEND
```

Figure 6-39 Submit JCL

5. After the job runs, return to the Observation List panel in APA and note that the observation was created. In our scenario, it is request number 0634, as shown in Figure 6-40.

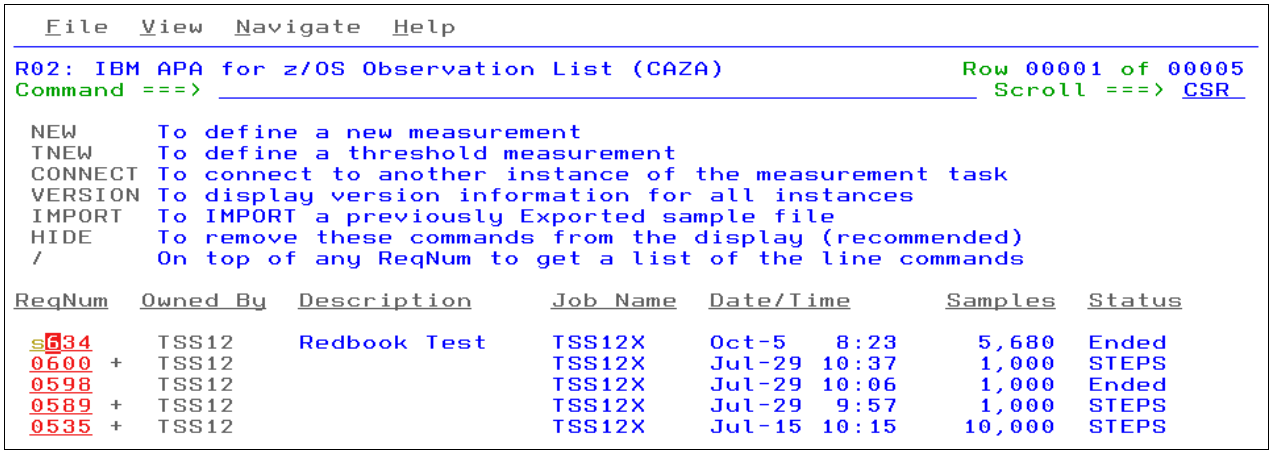


Figure 6-40 New observation that completed

6. To obtain the Performance Reports panel that is shown in Figure 6-41, enter **S** in the ReqNum field on the left side of this observation.

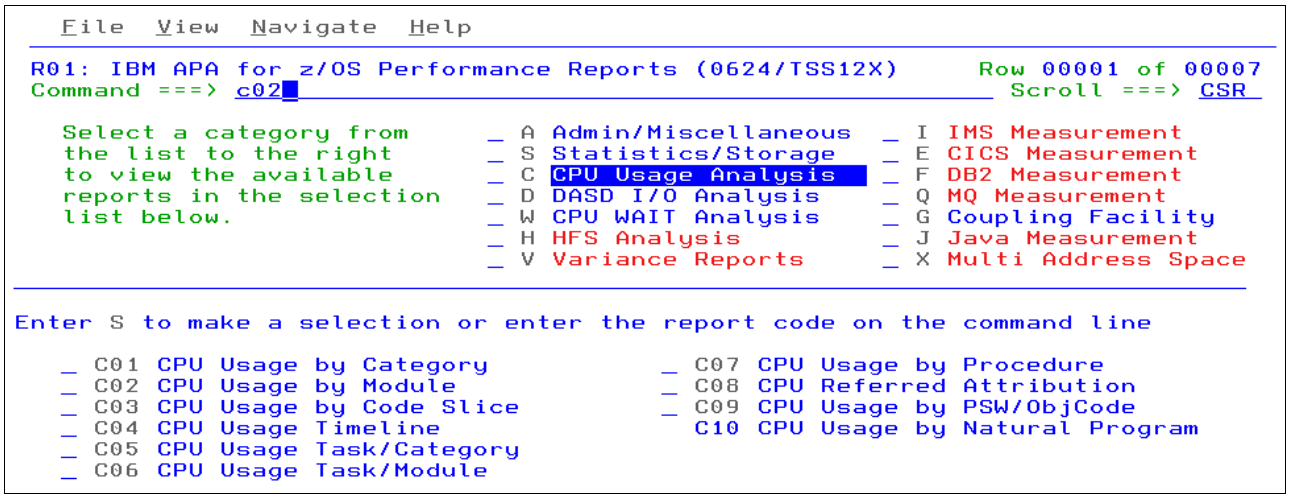


Figure 6-41 Results of s entered

From this panel, we have the choice of a number of APA analysis reports.

7. Select **CPU Usage by Module**.

   The CPU Usage by Module display is displayed, as shown in Figure 6-42.

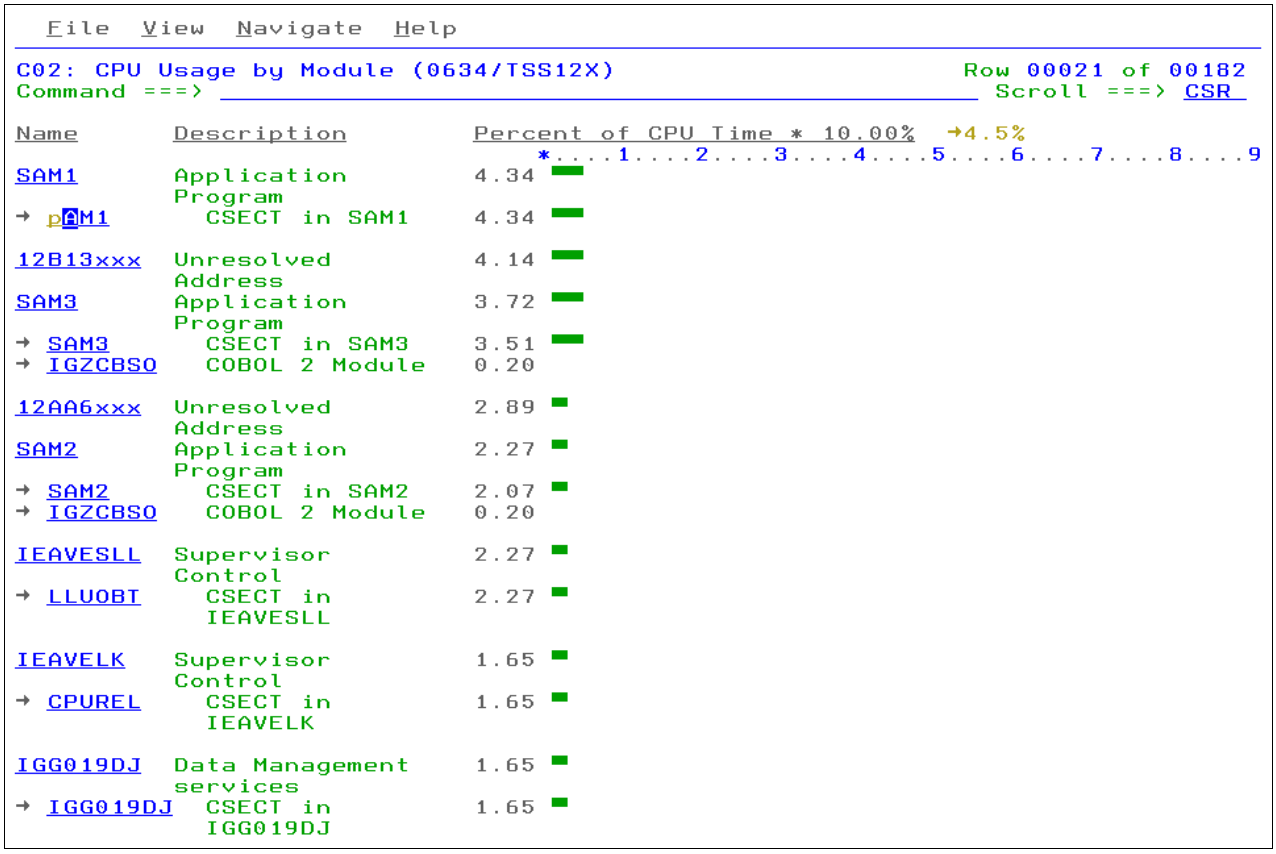


Figure 6-42 CPU usage by module

This report gives us an insight into how much CPU time was used by the various modules that were started when this program was run. For instance, we see that 4.34% of CPU time was spent in the main SAM1 module, 2.27% in the called program SAM2, and 3.72% in the other called program SAM3. The **P** line command (see Figure 6-43) added the source mapping to the C02 report, thus making it easy to see the lines of code the program spent the most time running.

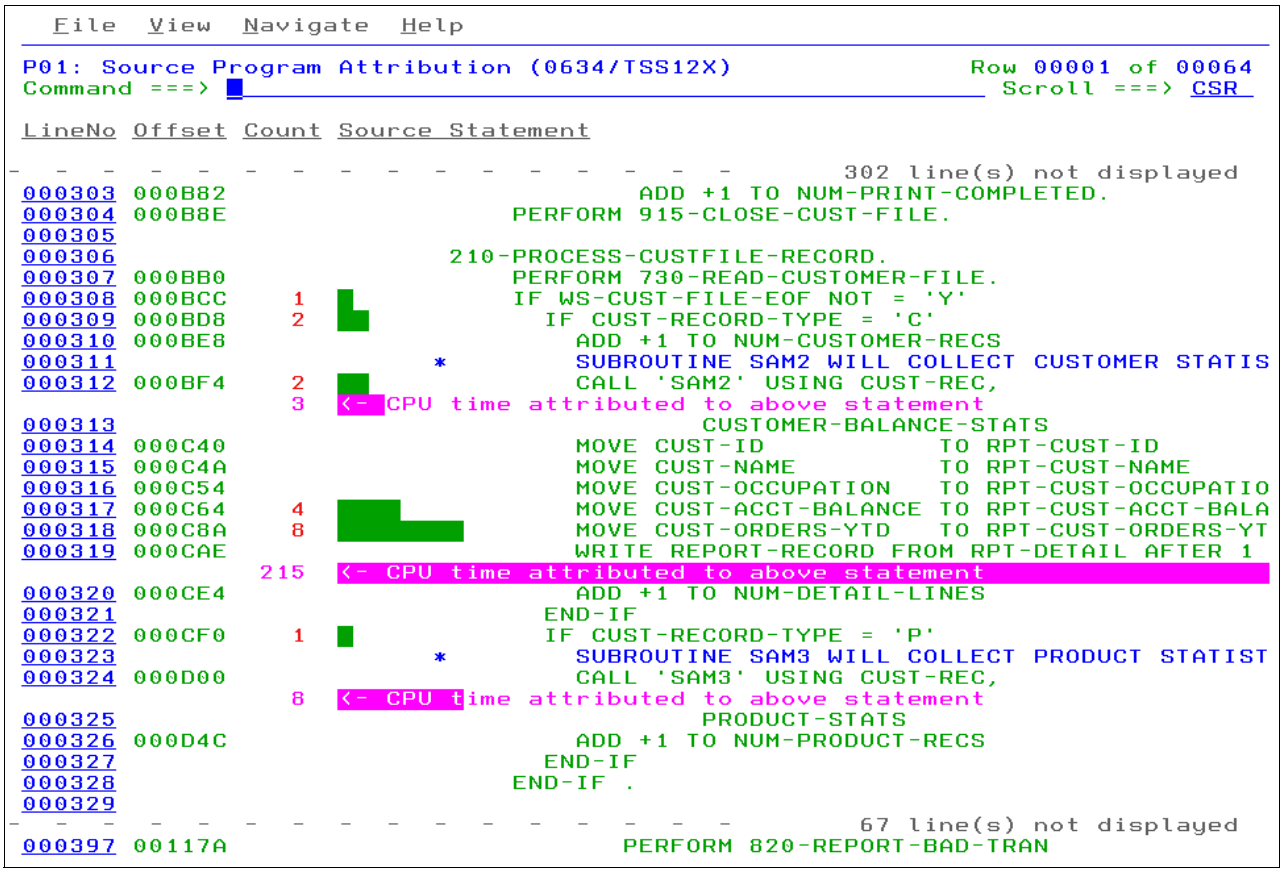


Figure 6-43 C02 report with source mapping

6.4.2 Starting APA for a batch program from CICS Explorer

Before the batch job is submitted, we must create an observation request to identify to APA the batch job that we are interested in analyzing.

Complete the following steps to start APA for a batch program from CICS Explorer:

1. In CICS Explorer, select the **APA/GUI** perspective, as shown in Figure 6-44.

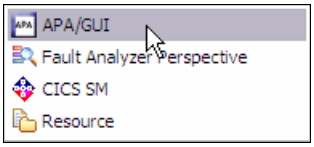


Figure 6-44 APA/GUI from CICS Explorer

2. Click the **Refresh** icon to refresh the list of observations that are available for this connection, as shown in Figure 6-45.

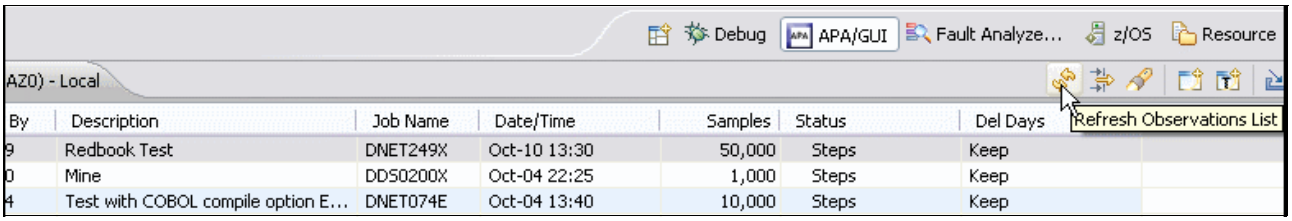


Figure 6-45 Refresh the observation list

We now see the observation that was created for our scenario at the top of the list, as shown in Figure 6-46.

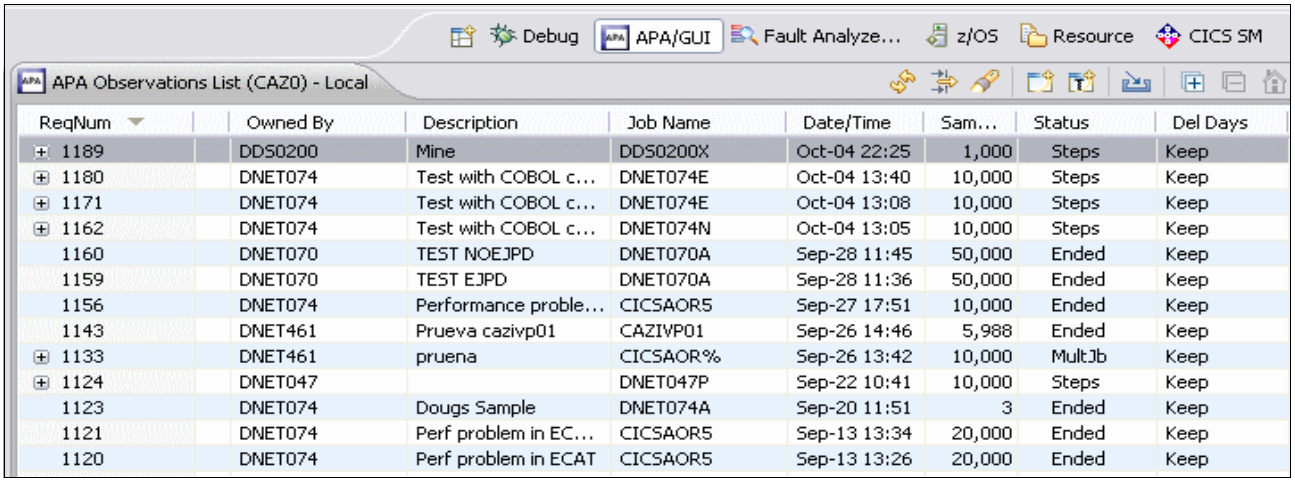


Figure 6-46 New observation is displayed

3. To see the reports that were created for this observation, right-click the observation and select **Download Reports** from the pull-down menu, as shown in Figure 6-47.

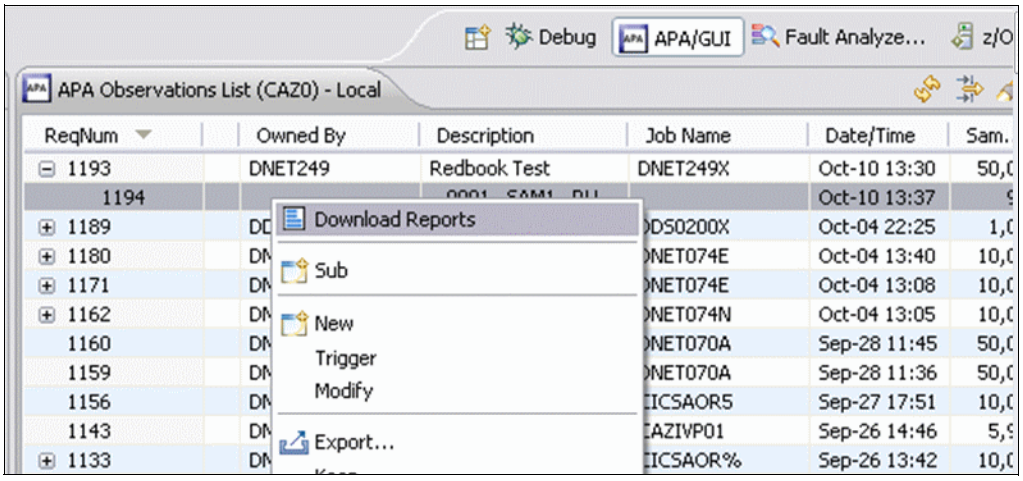


Figure 6-47 Download reports

An icon is displayed to the left of the observation to indicate that the reports for that observation were downloaded and are available for viewing from the APA perspective.

The details of this observation are displayed beneath the observation list, under the Details tab. However, we are interested in the reports.

4. Click the **Reports** tab.

We have the option of viewing any of the APA reports that were generated for this observation.

5. In this case, we select the **CPU Usage by Module** report, as shown in Figure 6-48.

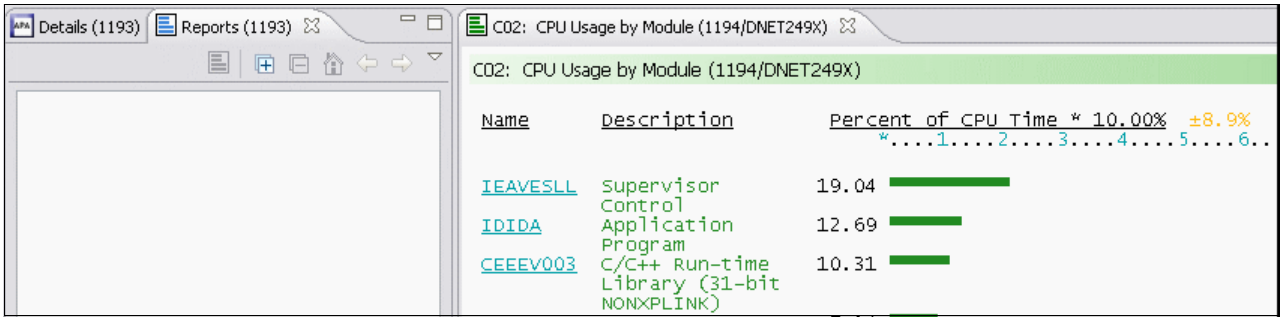


Figure 6-48 CPU Usage by Module

7

File Manager with Batch

This chapter introduces File Manager Base Feature batch processing techniques. Some of the most common processes are covered. File Manager interactive panels can be used to generate batch JCL.

It is important to note that during interactive JCL generation, any new data set is pre-allocated before the JCL is displayed. Also, when the interactive panels are used, data set names are generated in the control cards.

The focus of this chapter is to demonstrate the use of File Manager in static JCL. File Manager Functions and FASTREXX Functions are used. Some input files and examples are from the PD Tools Mentor Workshop ADLAB data sets that are used for hands-on training.

7.1 Background information

The most current *File Manager User's Guide and Reference* is available for download at the following website:

http://www-01.ibm.com/software/awdtools/filemanager/library/

Become familiar with Chapter 12, Chapter 13, and especially Chapter 16. Chapter 16 covers File Manager Functions and FASTREXX Functions in depth, including complete syntax, performance tips, options, return codes, related functions, and examples.

7.2 File Manager Functions and FASTREXX Functions

The File Manager normal processing can be enhanced by DFSORT, REXX, and FASTREXX (a subset of REXX) statements. For example, the normal processing of File Manager can be enhanced by using a template that contains REXX or FASTREXX statements. The following sections focus on File Manager Functions that can be enhanced by using FASTREXX Functions in JCL in-stream in a PROC.

7.2.1 File Manager Functions that can be enhanced

File Manager Functions are processed from SYSIN. Each function begins with $$FILEM followed by the function and any control statements that are required. One execution of File Manager can process several functions. When a PROC is used, they are separated with the /+ statement.

As shown in Figure 7-1, the data set copy (DSC) default data definitions (DDs) are used: DDIN and DDOUT. The first data set print (DSP) uses the default DD of DDIN. The second DSP is redirected to DDOUT for its input. File Manager copies ADLAB.CUST1 to ADLAB.CUST1.COPY1 and prints the input and output files, all from one execution.

```
//STEP001 EXEC PGM=FMNMAIN
//SYSPRINT DD  SYSOUT=*
//DDIN     DD DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
//DDOUT    DD DISP=SHR,DSN=&SYSUID..ADLAB.CUST1.COPY1
//SYSIN    DD *
$$FILEM DSC
$$FILEM DSP
$$FILEM DSP INPUT=DDOUT
```

Figure 7-1 DSC and two DSP

The following File Manager functions can be used in batch jobs, TSO, REXX and CLIST procedures:

- DSC

  DSC can be enhanced with REXX and DFSORT.

- Data Set Edit Batch

  Data Set Edit Batch (DSEB) can be enhanced only by REXX and control cards that are required.

- DSP

  DSP can be enhanced with REXX and DFSORT.

- Data Set Update

  Data Set Update (DSU) can be enhanced only by REXX and control cards that are required.

- Find or Change

  Find or Change (FCH) can be enhanced only by REXX and control cards that are optional.

7.2.2 FASTREXX functions used to enhance File Manager functions

There are 48 External REXX Functions available, of which 43 are FASTREXX Functions. The following functions are most commonly used:

- CHG_OUT

  Changes a character string in the output record.

- FLD

  Refers to a field from the current input record.

- FLD_CO

  Searches a field in the input record for one or more occurrences of a string, or tests a field in the input record for one or more numeric values.

- FLD_OUT

  Overlays the output record with a field from the input record.

- FLDI

  Performs a conditional test against an input record field.

- FLDO

  Performs a conditional test against an output record field.

- OVLY_OUT

  Overlays the output record with a literal (constant) or variable value.

- TALLY

  Totals a field value and reports the total.

- WRITE

  Explicitly writes a record.

7.2.3 Examples to enhance File Manager functions with FASTREXX functions

We now combine the File Manager functions that can be enhanced with the nine most commonly used FASTREXX functions.

Each of the following JCL example figures includes a text box that contains a ruler, the File Manager function, and the FASTREXX function. By using these features, you can highlight and capture the text then paste the text in a PDS(E) member, which should eliminate typographical errors.

DSC and CHG_OUT

The example that is shown in Figure 7-2 changes all occurrences of the string “Holly” to “Hollie”, starting in position 1 to the end of the record.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00001) - 01.02        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000010 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000020 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000100 //*
000110 //STEP001 EXEC PGM=IEFBR14
000120 //DD1      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000130 //             DSN=&SYSUID..ADLAB.CUST1.COPY1
000140 //*
000200 //STEP002 EXEC PGM=FMNMAIN
000300 //SYSOUT   DD  SYSOUT=*
000400 //SYSPRINT DD  SYSOUT=*
000500 //FMNTSPRT DD  SYSOUT=*
000600 //FMNSRTP  DD  SYSOUT=*
000700 //SYSTERM  DD  SYSOUT=*
000900 //DDIN     DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001000 //DDOUT    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001100 //             DCB=*.DDIN,DSN=&SYSUID..ADLAB.CUST1.COPY1
001200 //SYSIN    DD  *
001300 $$FILEM DSC PROC=*
001500  CHG_OUT('Holly','Hollie',0,1,0,'')
001600 /+
****** ***************************** Bottom of Data *****************************
```

Figure 7-2 DSC and CHG_OUT

Figure 7-3 shows File Manager function DSC and the FASTREXX function CHG_OUT are being processed in-stream by the SYSIN DD * statement. The /+ ends the PROC, so multiple PROCs can be passed to one execution of File Manager. The syntax changes the string “Holly” to “Hollie” and includes the following conditions:

- 0 = all occurrences
- 1 = start position
- 0 = end of the record
- The ‘ ‘ parameter (which is the default) says that the function behaves without text sensitivity.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM DSC PROC=*
 CHG_OUT('Holly','Hollie',0,1,0,'')
/+
```

Figure 7-3 DSC and CHG_OUT

Figure 7-4 shows the file before the CHG_OUT is issued. We issued a Find All HOLL command and the results are three strings of HOLLY were found.

```
 Process    Options    Help
View              TSS12.ADLAB.CUST1                          3 string(s) found
Command ===> ____________________________________________________ Scroll CSR
                                        Record AT TOP        Format TABL
       CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                             #5         #6 #7
       AN 1:5  AN 6:17                  PD 23:5     BI 28:2 AN 30:20
       <--->   <---+----1----+->    <---+----1>     <---+>  <---+----1----+---->
****** ****  Top of data  ****
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 41 Line(s) excluded
000042 33333   Graham, Holly                0.00          1 3100 Oaktree Ct
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 4 Line(s) excluded
000047 40045   Wood, Holly              25000.02          7 90210 Mt. Lee Rd
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 20 Line(s) excluded
000068 60606   Holly, Aretha              458.84          2 52 Thayer Street
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 32 Line(s) excluded
****** ****  End of data  ****
```

Figure 7-4 Before CHG_OUT

Figure 7-5 shows the new ADLAB.CUST1.COPY1 data set after the CHG_OUT function is used. Again, the Find ALL HOLL command was used and the results are three strings of HOLLIE were found.

```
 Process    Options    Help
View              TSS12.ADLAB.CUST1.COPY1                    3 string(s) found
Command ===> ____________________________________________________ Scroll CSR
                                        Record AT TOP        Format TABL
       CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                             #5         #6 #7
       AN 1:5  AN 6:17                  PD 23:5     BI 28:2 AN 30:20
       <--->   <---+----1----+->    <---+----1>     <---+>  <---+----1----+---->
****** ****  Top of data  ****
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 41 Line(s) excluded
000042 33333   Graham, Hollie               0.00          1 3100 Oaktree Ct
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 4 Line(s) excluded
000047 40045   Wood, Hollie             25000.02          7 90210 Mt. Lee Rd
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 20 Line(s) excluded
000068 60606   Hollie, Aretha             458.84          2 52 Thayer Street
- - -  - - - - - - - - - - - - - - - - - - - - - - - - 32 Line(s) excluded
****** ****  End of data  ****
```

Figure 7-5 After CHG_OUT

DSP and FLD

The example that is shown in Figure 7-6 prints the third record that contains the string “USA” and every third string thereafter, up to a total of three printed records.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00002) - 01.02        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000210 //*
000300 //STEP001 EXEC PGM=FMNMAIN
000400 //SYSOUT    DD  SYSOUT=*
000500 //SYSPRINT  DD  SYSOUT=*
000600 //FMNTSPRT  DD  SYSOUT=*
000700 //FMNSRTP   DD  SYSOUT=*
000800 //SYSTERM   DD  SYSOUT=*
000900 //DD01      DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001000 //SYSIN     DD  *
001010 $$FILEM SET HEADERPG=NO,PAGESIZE=60,PRTTRANS=ON,PRINTLEN=80
001100 $$FILEM DSP  INPUT=DD01,
001200 $$FILEM      NLRECS=3,
001300 $$FILEM      PROC=*
001400  IF FLD(66,3) == 'USA'
001500  THEN DO
001600   SETN(S_CNT,'+1')                  /* INCREMENT SELECT COUNT          */
001700   IF TESTN(S_CNT,'\=','3')          /* REACHED SELECT COUNT ?          */
001800   THEN RETURN DROP                  /* NO, DROP RECORD                 */
001900   SETN(S_CNT,'0')                   /* RESET SELECT COUNT              */
002000   RETURN                            /* PROCESS THE SELECTED RECORD     */
002100  END
002200  RETURN DROP                        /* NO CRITERIA MATCH, DROP         */
002300 /+
****** **************************** Bottom of Data ******************************
```

Figure 7-6 DSP and FLD

Figure 7-7 shows File Manager function DSP and the FASTREXX function FLD are being used to search the record, starting in position 66 for a length of three spaces. If the search is successful, the PROC uses a FASTREXX variable to track the occurrences of the string. The number of records to print is controlled by the $$FILEM control statement NLRECS=3.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM SET HEADERPG=NO,PAGESIZE=60,PRTTRANS=ON,PRINTLEN=80
$$FILEM DSP  INPUT=DD01,
$$FILEM      NLRECS=3,
$$FILEM      PROC=*
 IF FLD(66,3) == 'USA'
 THEN DO
  SETN(S_CNT,'+1')
  IF TESTN(S_CNT,'\=','3')
  THEN RETURN DROP
  SETN(S_CNT,'0')
  RETURN
 END
 RETURN DROP
/+
```

Figure 7-7 DSP and FLD

The output in SDSF is shown in Figure 7-8. At the bottom of the report, the FMNBE082 message indicates that three records were printed.

```
  Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TSS12X   JOB02967  DSID   103 LINE 10      COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
  SETN(S_CNT,'+1')                 /* INCREMENT SELECT COUNT          */ 00160000
  IF TESTN(S_CNT,'\=','3')         /* REACHED SELECT COUNT ?          */ 00170000
  THEN RETURN DROP                 /* NO, DROP RECORD                 */ 00180000
  SETN(S_CNT,'0')                  /* RESET SELECT COUNT              */ 00190000
  RETURN                           /* PROCESS THE SELECTED RECORD     */ 00200000
 END                                                                     00210000
 RETURN DROP                       /* NO CRITERIA MATCH, DROP         */ 00220000

FMNBB061 REXX procedure statements processed by File Manager (FASTREXX).
* * * * Device 2826, 3390, VOLSER=SMS006                              * * * *
* *     Data Set Name: TSS12.ADLAB.CUST1                                  * *
* *     DSORG  PS, RECFM VB, LRECL 600 BLKSIZE 27998                      * *
Record Length: 331            02202Major, Art         %  1512 Pine Bluff    H
                              armon        MNUSA              &          <
                                  <    <         <   h     @    <College studen
                              t            e sample notes. More notes. And ye
                              t more notes. More notes. And yet more notes. Some
                               sample notes. One last note.

Record Length: 331            05570Parker, Ford         @  3039 Manning St.    D
                              earborn      MIUSA                          %
                                  %              <              <Garage attenda
                              nt            Some sample notes. More notes. And
                               yet more notes. More notes. And yet more notes. S
                              ome sample notes. One last note.

Record Length: 331            06711Dubree, Dustin         9229 Delegate's Row I
                              ndianapolis  INUSA              <       %
                                  <    <         %         %     Demolition Con
                              tractor      me sample notes. More notes. And y
                              et more notes.Some sample notes. More notes. And y
                              et more notes.Some sample notes. M


FMNBE082 3 record(s) printed
******************************** BOTTOM OF DATA ********************************
```

Figure 7-8   Three printed records

FCH and FLD_CO

The example that is shown in Figure 7-9 uses the FCH function to search a PDS(E) library for members that contain the string “this” and “that” (both strings must be on the same record) or the string “SAM1”. FCH searches for “this” and “that”. It also searches for “this” or “that”. To process the more complex criteria (in our example), the FASTREXX FLD_CO Function does the job.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 -------------------------------------------------------------------------------
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00003) - 01.02         Columns 00001 00072
Command ===>                                                    Scroll ===> CSR
****** ***************************** Top of Data ******************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000210 //*
001400 //STEP001 EXEC PGM=FMNMAIN
001500 //SYSPRINT DD SYSOUT=*
001600 //DDIN      DD DISP=SHR,DSN=&SYSUID..ADLAB.JCL
001800 //SYSIN     DD *
001900 $$FILEM FCH MEMBER=*,PROC=*
002000   IF (FLD_CO(,,U,'THIS') & FLD_CO(,,U,'THAT')) | ,
002100       FLD_CO(,,U,'PGM=SAM1') THEN ,
002200       RETURN PROCESS MEMBER
002300   ELSE RETURN DROP
002400 /+
****** **************************** Bottom of Data ****************************
```

Figure 7-9   Search for THIS and THAT or SAM1

Figure 7-10 shows File Manager function FCH and the FASTREXX function FLD_CO are being used with logic that processes the existence of two strings on one record or the existence of a different string on one record. All three strings also can exist on one record.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD *
$$FILEM FCH MEMBER=*,PROC=*
  IF (FLD_CO(,,U,'THIS') & FLD_CO(,,U,'THAT')) | ,
      FLD_CO(,,U,'PGM=SAM1') THEN ,
      RETURN PROCESS MEMBER
  ELSE RETURN DROP
/+
```

Figure 7-10 Search for THIS and THAT or SAM1

As shown in Figure 7-11, the report shows some of the members that contain the strings “THIS” and “THAT” or the string “SAM1. The member COBILINK has the strings “THIS” and “THAT”. The member FMBATFCH includes the strings “THIS”, “THAT”, and “SAM1”. The member FMDSP01 only has the string “SAM1”.

```
   Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TSS12X   JOB02968  DSID   102 LINE 22       COLS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> CSR
IBM File Manager for z/OS
$$FILEM FCH MEMBER=*,PROC=*                                          00190002
FMNBC514 REXX procedure statements read from SYSIN.

  IF (FLD_CO(,,U,'THIS') & FLD_CO(,,U,'THAT')) | ,                   00200002
      FLD_CO(,,U,'PGM=SAM1') THEN ,                                  00210002
      RETURN PROCESS MEMBER                                          00220002
  ELSE RETURN DROP                                                   00230002

FMNBB061 REXX procedure statements processed by File Manager (FASTREXX).

Find/Change Listing DSN: TSS12.ADLAB.JCL

 COBILINK                    ----- STRING(S) FOUND/CHANGED -----
Record Number
        7s //*    THIS CAN BE USED TO PREPARE OS/VS COBOL PROGRAMS THAT RUN

 FMBATFCH                    ----- STRING(S) FOUND/CHANGED -----
Record Number
        5s //* THIS STEP ITENTIFIES OCCURRENCES OF 'THIS' AND 'THAT' OR 'SAM1 '
       14s IF (FLD_CO(,,U,'THIS') & FLD_CO(,,U,'THAT')) ,
       15s     FLD_CO(,,U,'PGM=SAM1 ') THEN ,

 FMDSP01                     ----- STRING(S) FOUND/CHANGED -----
Record Number
        7s //*  THIS EXAMPLE PRINTS JCL MEMBERS THAT EXECUTE A CERTAIN PROGRAM
       15s    FLD_CO(,,U,'PGM=SAM1') THEN RETURN PROCESS MEMBER;

 EMECH00                     ----- STRING(S) FOUND/CHANGED -----
```

Figure 7-11 Members with the strings

DSC and FLD_OUT

The example that is shown in Figure 7-12 copies the input data set while a 5-byte field is written, starting in position 77 out to position 85. No other fields are changed in the output data set. So, this example copies the data and moves a field into another position over the existing data.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00004) - 01.00        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ******************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000300 //*
000400 //STEP001 EXEC PGM=IEFBR14
000500 //DD1      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000600 //              DSN=&SYSUID..ADLAB.CUST1.COPY1
000700 //*
000800 //STEP002 EXEC PGM=FMNMAIN
000900 //SYSOUT   DD  SYSOUT=*
001000 //SYSPRINT DD  SYSOUT=*
001100 //FMNTSPRT DD  SYSOUT=*
001200 //FMNSRTP  DD  SYSOUT=*
001300 //SYSTERM  DD  SYSOUT=*
001400 //DDIN     DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001500 //DDOUT    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001600 //              DCB=*.DDIN,DSN=&SYSUID..ADLAB.CUST1.COPY1
001700 //SYSIN    DD  *
001800 $$FILEM DSC PROC=*
001900  FLD_OUT(77,5,82,5)
002000 /+
****** **************************** Bottom of Data ****************************
```

Figure 7-12 DSC and FLD_OUT

Figure 7-13 shows the File Manager function DSC and the FASTREXX function FLD_OUT are being used to copy all of the data in the input file to the new output file. It overlays the data in position 85 for a length of 5 bytes with the same data in position 77 for a length of 5 bytes.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM DSC PROC=*
 FLD_OUT(77,5,82,5)
/+
```

Figure 7-13 DSC and FLD_OUT

Figure 7-14 shows the values in the two fields that are referenced by #11(1) and #11(2).

```
 Process    Options    Help
View              TSS12.ADLAB.CUST1                          Top of 100
Command ===> _                                               Scroll CSR
                                                 Record AT TOP  Format TABL
          MONTH(1)     MONTH(2)     MONTH(3)     MONTH(4)     MONTH(5)     MONTH(6)
             #11          #11          #11          #11          #11          #11
           PD 77:5      PD 82:5      PD 87:5      PD 92:5      PD 97:5     PD 102:5
       <---+----1> <---+----1> <---+----1> <---+----1> <---+----1> <---+----1>
****** ****  Top of data  ****
000001         4.84         1.00         3.00        10.89         5.00        10.89
000002         1.21         1.21         1.21         1.21         1.21         1.21
000003         3.63        15.00        10.89         4.84         7.00         4.84
```

Figure 7-14 Before FLD_OUT

Figure 7-15 shows that the values in #11(2) were overlaid with the values in #11(1). All other fields remain as they are in the input data set.

```
 Process    Options    Help
------------------------------------------------------------------------------
View                TSS12.ADLAB.CUST1.COPY1                    Top of 100
Command ===> _____________________________________________________ Scroll CSR
                                                    Record AT TOP  Format TABL
           MONTH(1)     MONTH(2)     MONTH(3)     MONTH(4)     MONTH(5)     MONTH(6)
              #11          #11          #11          #11          #11          #11
            PD 77:5      PD 82:5      PD 87:5      PD 92:5      PD 97:5     PD 102:5
         <---+----1> <---+----1> <---+----1> <---+----1> <---+----1> <---+----1>
****** ****  Top of data  ****
000001          4.84         4.84         3.00        10.89         5.00        10.89
000002          1.21         1.21         1.21         1.21         1.21         1.21
000003          3.63         3.63        10.89         4.84         7.00         4.84
```

Figure 7-15 After FLD_OUT

DSC and FLDI

The example that is shown in Figure 7-16 searches for a record that contains strings “HOLLY” and “Wood”. The CU function uppercases the search so the string is found regardless of the case that is specified. The CO function is case-sensitive and matches only what is coded (by case).

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Iest  Help
------------------------------------------------------------------------------
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00005) - 01.01        Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000300 //*
000400 //STEP001 EXEC PGM=IEFBR14
000500 //DD1      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000600 //             DSN=&SYSUID..ADLAB.CUST1.COPY1
000700 //*
000800 //STEP002 EXEC PGM=FMNMAIN
000900 //SYSOUT   DD  SYSOUT=*
001000 //SYSPRINT DD  SYSOUT=*
001100 //FMNTSPRT DD  SYSOUT=*
001200 //FMNSRTP  DD  SYSOUT=*
001300 //SYSTERM  DD  SYSOUT=*
001500 //DD01     DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001600 //DD01O    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001700 //             DCB=*.DD01,DSN=&SYSUID..ADLAB.CUST1.COPY1
001800 //SYSIN    DD  *
001900 $$FILEM DSC  INPUT=DD01,
002000 $$FILEM      OUTPUT=DD01O,
002100 $$FILEM      PROC=*
002200  IF FLDI(1,0,C,'CU','HOLLY'),       /* CU is case insensitive         */
002300   & FLDI(1,0,C,'CO','Wood')         /* CO is case sensitive           */
002400  THEN DO
002500   RETURN                            /* process record                 */
002600  END
002700  RETURN DROP                        /* no criteria match, drop        */
002800 /+
****** ***************************** Bottom of Data ***************************
```

Figure 7-16 DSC and FLDII

Figure 7-17 shows File Manager function DSC and FASTREXX function FLDI are being used to copy a record that contains the upper-cased contains function search string “HOLLY” and the mixed-case contains function search string “Wood”. If both strings are found, the record is written to the DD01O. The File Manager default DDs for DSC are DDIN and DDOUT. You can use other DD names by redirecting INPUT and OUTPUT.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM DSC  INPUT=DD01,
$$FILEM      OUTPUT=DD01O,
$$FILEM      PROC=*
 IF FLDI(1,0,C,'CU','HOLLY'),
  & FLDI(1,0,C,'CO','Wood')
 THEN DO
  RETURN
 END
 RETURN DROP
/+
```

Figure 7-17 DSC and FLDI

As shown in Figure 7-18, the input file shows record number 47 (at the top of the display) of 100 records.

```
 Process    Options    Help
--------------------------------------------------------------------------------
View                 TSS12.ADLAB.CUST1                        Rec 47 of 100
Command ===> _                                                 Scroll CSR
                                                 Record 47     Format TABL
       CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                             #5          #6 #7
       AN 1:5  AN 6:17                     PD 23:5     BI 28:2 AN 30:20
       <---->  <---+----1----+->    <---+----1>      <---+> <---+----1----+---->
000047 40045   Wood, Holly              25000.02           7 90210 Mt. Lee Rd
000048 40909   Burr, Tim                 7766.55           0 4111 Northside PkWay
```

Figure 7-18 Record number 47

The output file in Figure 7-19 contains the one record that met the search criteria.

```
 Process    Options    Help
--------------------------------------------------------------------------------
View                 TSS12.ADLAB.CUST1.COPY1                  Top of 1
Command ===> _                                                 Scroll CSR
                                                 Record AT TOP Format TABL
       CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                             #5          #6 #7
       AN 1:5  AN 6:17                     PD 23:5     BI 28:2 AN 30:20
       <---->  <---+----1----+->    <---+----1>      <---+> <---+----1----+---->
****** ****  Top of data  ****
000001 40045   Wood, Holly              25000.02           7 90210 Mt. Lee Rd
****** ****  End of data  ****
```

Figure 7-19 One copied record

DSC with FLDO and OVLY_OUT

The example that is shown in Figure 7-20 copies all of the records in the input file to an output file and overlays one of two strings into a field that is based on a string value are being found.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00006) - 01.01        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000300 //*
000400 //STEP001 EXEC PGM=IEFBR14
000500 //DD1      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000600 //              DSN=&SYSUID..ADLAB.CUST1.COPY1
000700 //*
000800 //STEP002 EXEC PGM=FMNMAIN
000900 //SYSOUT   DD  SYSOUT=*
001000 //SYSPRINT DD  SYSOUT=*
001100 //FMNTSPRT DD  SYSOUT=*
001200 //FMNSRTP  DD  SYSOUT=*
001300 //SYSTERM  DD  SYSOUT=*
001500 //DD01     DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001600 //DD01O    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001700 //              DCB=*.DD01,DSN=&SYSUID..ADLAB.CUST1.COPY1
001800 //SYSIN    DD  *
001900 $$FILEM DSC  INPUT=DD01,
002000 $$FILEM      OUTPUT=DD01O,
002100 $$FILEM      PROC=*
002200  IF FLDO(1,0,C,'CO','1001')      /* using valid value for CUST1 file */
002300  THEN DO
002400   OVLY_OUT('YYYY',1)
002500  END
002600  ELSE DO
002700   OVLY_OUT('ZZZZ',1)
002800  END
002900 /+
****** **************************** Bottom of Data ******************************
```

Figure 7-20 Copy records

Figure 7-21 shows File Manager function DSC and FASTREXX function OVLY_OUT are being used with the FLDO FASTREXX function. FLDO checks if the record contains the string “1001” anywhere in the record. If found, OVLY_OUT writes the string “YYYY” starting in position 1 of the output record; otherwise, the string “ZZZZ” is written. Both functions can specify a starting position and length.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM DSC  INPUT=DD01,
$$FILEM      OUTPUT=DD01O,
$$FILEM      PROC=*
 IF FLDO(1,0,C,'CO','1001')
 THEN DO
  OVLY_OUT('YYYY',1)
 END
 ELSE DO
  OVLY_OUT('ZZZZ',1)
 END
/+
```

Figure 7-21 File Manager Function DSC

The input file is shown in Figure 7-22. Record 1 contains the string “1001” in the CUST-ID field.

```
 Process    Options    Help
------------------------------------------------------------------------------
View                  TSS12.ADLAB.CUST1                        Top of 100
Command ===> _                                                 Scroll CSR
                                                 Record AT TOP  Format TABL
        CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
        #3      #4                          #5          #6 #7
        AN 1:5  AN 6:17                  PD 23:5     BI 28:2 AN 30:20
        <--->   <---+----1----+->   <---+----1>      <---+> <---+----1----+---->
****** ****  Top of data  ****
000001 01001   Lynn, Amanda              610.05          10 89 Clay Springs Rd
000002 02200   Graham, Anna               67.68           9 119 North Lake Road
000003 02202   Major, Art               1234.56           5 1512 Pine Bluff
000004 03003   Prentice, Anna              0.00           7 33 Renshaw
000005 03390   Deeds, Darren              74.00           3 649 Brown Street
```

Figure 7-22 Input file

The output file is shown in Figure 7-23. Record 1 now contains the CUST-ID of “YYYY1” and all other records contain “ZZZZ”, followed by a single number.

```
 Process    Options    Help
------------------------------------------------------------------------------
View                  TSS12.ADLAB.CUST1.COPY1                  Top of 100
Command ===> _                                                 Scroll CSR
                                                 Record AT TOP  Format TABL
        CUST-ID NAME                 ACCT-BALANCE ORDERS-YTD ADDR
        #3      #4                          #5          #6 #7
        AN 1:5  AN 6:17                  PD 23:5     BI 28:2 AN 30:20
        <--->   <---+----1----+->   <---+----1>      <---+> <---+----1----+---->
****** ****  Top of data  ****
000001 YYYY1   Lynn, Amanda              610.05          10 89 Clay Springs Rd
000002 ZZZZ0   Graham, Anna               67.68           9 119 North Lake Road
000003 ZZZZ2   Major, Art               1234.56           5 1512 Pine Bluff
000004 ZZZZ3   Prentice, Anna              0.00           7 33 Renshaw
```

Figure 7-23 Output

DSU and DSEB with TALLY

The example that is shown in Figure 7-24 shows two File Manager functions: DSU and DSEB and FASTREXX function TALLY are used to total two columns of numbers.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00007) - 01.02        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000300 //*
000400 //STEP001 EXEC PGM=FMNMAIN
000500 //SYSPRINT DD SYSOUT=*
000600 //DDIN     DD DISP=SHR,DSN=&SYSUID..ADLAB.CUST1.COPY1
000900 //SYSIN    DD *
002100 $$FILEM DSU PROC=*
002300    TALLY(23,5,P,'ACCT-BALANCE')
002400    TALLY(28,2,B,'ORDERS-YTD')
002410 /+
002500 $$FILEM DSEB PROC=*
002600   CHECKEOF = 0
002610   DO WHILE CHECKEOF /= 'EOF'
002700    TALLY(23,5,P,'ACCT-BALANCE')
002800    TALLY(28,2,B,'ORDERS-YTD')
002900   CHECKEOF = DOWN(1)
003000   END
003100 /+
****** **************************** Bottom of Data ******************************
```

Figure 7-24 DSU and DSEB with TALLY

Figure 7-25 shows File Manager functions DSU and FASTREXX function TALLY are being used to total two columns of numbers. The ACCT-BALANCE field is packed decimal and the ORDERS-YTD is a binary field. DSU often is used for in-place updates, but in this case, it gives the tally. It is best not to use DSP with TALLY unless you want to print the file and total the columns.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
$$FILEM DSU PROC=*
   TALLY(23,5,P,'ACCT-BALANCE')
   TALLY(28,2,B,'ORDERS-YTD')
```

Figure 7-25 DSU and TALLY

The text box in Figure 7-26 shows that File Manager functions DSEB and FASTREXX function TALLY are being used to total two columns of numbers. The ACCT-BALANCE field is packed decimal and the ORDERS-YTD is a binary field. DSEB often is used for in-place updates, but in this case, it gives the tally.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
$$FILEM DSEB PROC=*
  CHECKEOF = 0
  DO WHILE CHECKEOF /= 'EOF'
   TALLY(23,5,P,'ACCT-BALANCE')
   TALLY(28,2,B,'ORDERS-YTD')
  CHECKEOF = DOWN(1)
  END
```

Figure 7-26 DSEB and TALLY

DSEB differs from DSU in that it requires all of the I/O, which means that we set up a DO loop and based that loop on reaching the end-of-file (or not) and the DOWN(1) function gets the next record to continue processing. It is best not to use DSP with TALLY unless you want to print the file and total the columns.

As shown in Figure 7-27, the TALLY results are the same for DSU and DSEB. Both File Manager functions read 100 records.

```
   Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TSS12X   JOB03332  DSID   102 LINE 21       COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR

IBM File Manager for z/OS
$$FILEM DSU PROC=*                                                      00210001
FMNBC514 REXX procedure statements read from SYSIN.

   TALLY(23,5,P,'ACCT-BALANCE')                                         00230000
   TALLY(28,2,B,'ORDERS-YTD')                                           00240000

FMNBB061 REXX procedure statements processed by File Manager (FASTREXX).
FMNBB283 100 record(s) read from input data set
FMNBB434 0 record(s) updated
IBM File Manager for z/OS
TALLY summary report
------------------------------------------------------------------------------
ACCT-BALANCE     27970232
ORDERS-YTD            350
IBM File Manager for z/OS
$$FILEM DSEB PROC=*                                                     00250001
FMNBC514 REXX procedure statements read from SYSIN.

  CHECKEOF = 0                                                          00260001
  DO WHILE CHECKEOF /= 'EOF'                                            00261001
   TALLY(23,5,P,'ACCT-BALANCE')                                         00270001
   TALLY(28,2,B,'ORDERS-YTD')                                           00280001
  CHECKEOF = DOWN(1)                                                    00290001
  END                                                                   00300001

FMNBB060 REXX procedure statements processed by REXX.
FMNBB283 100 record(s) read from input data set
FMNBB434 0 record(s) updated
IBM File Manager for z/OS
TALLY summary report
------------------------------------------------------------------------------
ACCT-BALANCE     27970232
ORDERS-YTD            350
******************************** BOTTOM OF DATA ********************************
```

Figure 7-27 Results

DSC and WRITE

The example that is shown in Figure 7-28 shows File Manager function DSC and FASTREXX function WRITE are being used to split an input file into two output files.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       TSS12.REDBOOKS.JCL.CNTL(FMN00008) - 01.06        Columns 00001 00072
Command ===> _                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000100 //TSS12X JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000200 //           MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000300 //*
000400 //STEP001 EXEC PGM=IEFBR14
000500 //DD1      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000600 //              DSN=&SYSUID..ADLAB.CUST1.COPY1
000610 //DD2      DD  DISP=(MOD,DELETE),SPACE=(TRK,(1,1)),UNIT=SYSDA,
000620 //              DSN=&SYSUID..ADLAB.CUST1.COPY2
000700 //*
000800 //STEP002 EXEC PGM=FMNMAIN
000900 //SYSOUT   DD  SYSOUT=*
001000 //SYSPRINT DD  SYSOUT=*
001100 //FMNTSPRT DD  SYSOUT=*
001200 //FMNSRTP  DD  SYSOUT=*
001300 //SYSTERM  DD  SYSOUT=*
001500 //DD01     DD  DISP=SHR,DSN=&SYSUID..ADLAB.CUST1
001600 //DD01O    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001700 //              DCB=*.DD01,DSN=&SYSUID..ADLAB.CUST1.COPY1
001800 //DD02O    DD  DISP=(NEW,CATLG),SPACE=(CYL,(5,20),RLSE),
001810 //              DCB=*.DD01,DSN=&SYSUID..ADLAB.CUST1.COPY2
001900 //SYSIN    DD  *
002000 $$FILEM DSC  INPUT=DD01,
002100 $$FILEM      OUTPUT=DD01O,
002200 $$FILEM      PROC=*
002300  IF FLDI(1,0,C,'CO','01001','40045')   /* search for two values */
002400  THEN DO
002500   WRITE(DD01O)
002600  END
002700  ELSE DO
002800   WRITE(DD02O)
002900  END
003000  RETURN DROP
003100 /+
****** ***************************** Bottom of Data *****************************
```

Figure 7-28 Function DSC and FASTREXX Function WRITE

Figure 7-29 shows File Manager function DSC and FASTREXX function WRITE are being used with the FASTREXX function FLDI. FLDI is used to check the record for one of two strings, “01001” or “40045”. If one of the strings is found, the record is written to DD01O; otherwise, the record is written to DD02O.

```
//*-+----1----+----2----+----3----+----4----+----5----+----6----+----7--
//SYSIN    DD  *
$$FILEM DSC  INPUT=DD01,
$$FILEM      OUTPUT=DD01O,
$$FILEM      PROC=*
 IF FLDI(1,0,C,'CO','01001','40045')
 THEN DO
  WRITE(DD01O)
 END
 ELSE DO
  WRITE(DD02O)
 END
 RETURN DROP
/+
```

Figure 7-29 DSC function

We used File Manager Catalog Services Option 3.4 (see Figure 7-30) to list the input data set and the two output data sets.

```
 Process    Options    Help
 
 File Manager                  Data Set List                 Row 00001 of 00003
Command ===>                                                       Scroll CSR
Catalog ID ''                                                    Types ALL
        Data Set Name                                  Type   Volume   MV Creat →
        TSS12.ADLAB.CUST1.**                           *      *        *  *
_______ TSS12.ADLAB.CUST1                              NVSAM  SMS006      2011.
_______ TSS12.ADLAB.CUST1.COPY1                        NVSAM  SMS010      2011.
_______ TSS12.ADLAB.CUST1.COPY2                        NVSAM  SMS011      2011.
**** End of data ****
```

Figure 7-30 Option 3.4 from File Manager

Figure 7-31 shows only the records that contain the strings “01001” or “40045”.

```
 Process    Options    Help
 
View             TSS12.ADLAB.CUST1.COPY1                            Top of 2
Command ===>                                                        Scroll CSR
                                                  Record AT TOP     Format TABL
       CUST-ID NAME              ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                          #5         #6 #7
       AN 1:5  AN 6:17              PD 23:5    BI 28:2 AN 30:20
       <--->   <---+----1----+->  <---+----1>     <---+> <---+----1----+---->
****** ****  Top of data  ****
000001 01001   Lynn, Amanda            610.05         10 89 Clay Springs Rd
000002 40045   Wood, Holly           25000.02          7 90210 Mt. Lee Rd
****** ****  End of data  ****
```

Figure 7-31 Shows results

Figure 7-32 shows all 98 records that did not contain the strings “01001” or “40045”.

```
 Process    Options    Help
 
View             TSS12.ADLAB.CUST1.COPY2                         Top of 98
Command ===>                                                        Scroll CSR
                                                  Record AT TOP     Format TABL
       CUST-ID NAME              ACCT-BALANCE ORDERS-YTD ADDR
       #3      #4                          #5         #6 #7
       AN 1:5  AN 6:17              PD 23:5    BI 28:2 AN 30:20
       <--->   <---+----1----+->  <---+----1>     <---+> <---+----1----+---->
****** ****  Top of data  ****
000001 02200   Graham, Anna             67.68          9 119 North Lake Road
000002 02202   Major, Art             1234.56          5 1512 Pine Bluff
```

Figure 7-32 All records

Be sure to use the text box examples that are presented in this chapter to cut and paste. You can try them in your own scenarios.

8

Manipulating z/OS resources by using File Manager

By using the File Manager Base component you can manipulate data that is stored in data sets, UNIX System Services files, WebSphere MQ queues, and various CICS resources[1]. You can edit and browse such resources by using File Manager's editor. You can also specify those various types of resources as a target to start File Manager utility functions online and batch. Currently, the following utility functions provide support:

- Find/Change (FCH)
- Data set create (DSG)
- Data set copy (DSC)
- Data set print (DSP)
- Data set compare (DSM)
- Data set update and data set edit batch (DSU/DSEB)

File Manager provides a set of useful capabilities when you are manipulating data that is stored in different resource types in a z/OS environment. The following example scenarios illustrate these capabilities:

- Edit and browse data that is stored in WebSphere MQ queues and CICS resources. You can format data by using a template.
- Copy test data that is stored in a data set to a WebSphere MQ queue.
- Copy test data that is stored in a data set to a CICS file, Transient Data queue, or Temporary Storage queue.
- Edit or browse a CICS resource or WebSphere MQ queue and save a copy of the data to a data set by using the SAVE AS action.
- Compare data that is stored in a CICS resource of a CICS system (such as a development environment) to a corresponding CICS resource in another CICS system (such as a test environment).

In this chapter, we demonstrate how to use File Manager to perform these scenarios under ISPF/batch environment, and how to use the File Manager plug-in for Eclipse.

[1] Support for CICS resources was added to File Manager Base component via APAR PM47010.

8.1 Data manipulation by using File Manager on the host

This section describes the use of File Manager for data manipulation.

8.1.1 Edit or browse resources by using File Manager editor

A File Manager editor entry panel is shown in Figure 8-1. You specify the name of the resource that you want to edit or browse along with other File Manager options. You can specify a data set name or a UNIX System Services file.

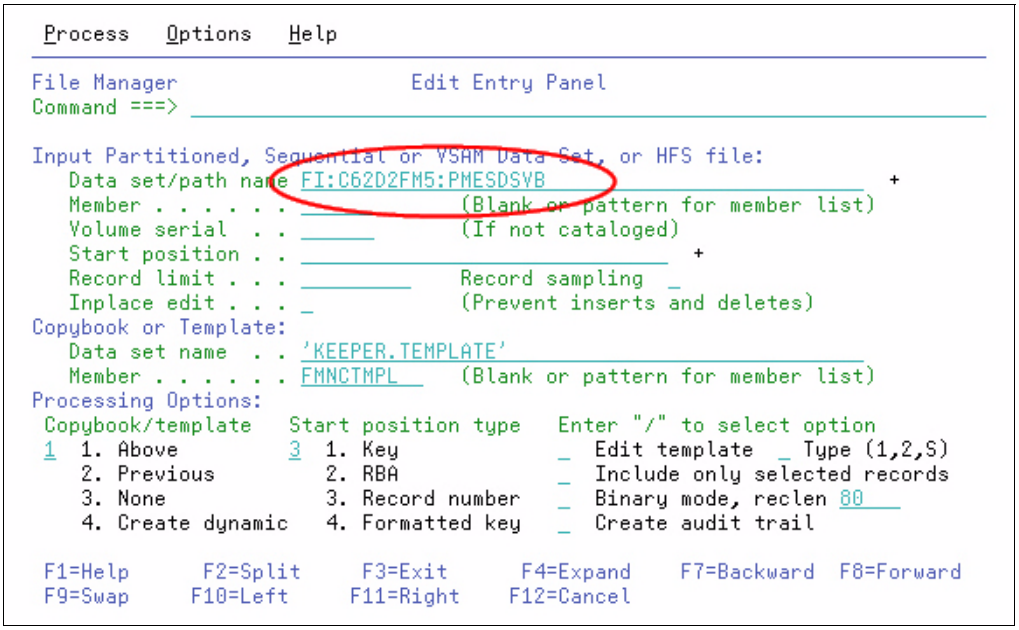


Figure 8-1 File Manager editor entry panel - specify different types of resources

You also can specify the following WebSphere MQ queue names and CICS resources:

- Specifying the WebSphere MQ:

  `MQ:manager-id:queue-name`

  - MQ: A prefix for specifying the WebSphere MQ queue.
  - manager-id: The WebSphere MQ manager that is to be used. If you specify a generic name, File Manager displays a list of matching managers to select from (such as MQ:*, which looks up all queue managers).
  - queue-name: The queue that is to be used. If you specify a generic name, File Manager displays a list of queues to select from (such as MQ:MQ72:*, which looks up all queues that are defined to queue manager MQ72).

- Specifying a CICS resource:

  `RT:APPLID:RNAME`

  – RT: Identifies CICS resource type. The following values are valid:
    - FI: CICS file
    - TD: Transient Data Queue
    - TS: Temporary Storage Queue

  – APPLID: VTAM applid for the CICS system. You can specify a generic name to display a list of CICS systems if an FMNCICS DD that is describing CICS applids is allocated to the ISPF session (such as FI:*, which looks up all CICS systems that are described by FMNCICS DD).

  – RNAME: Name of the resource. If you specify a generic name, File Manager displays a list of resource to select from (such as FI:C62D2FM5:*, which looks up all CICS files that are defined in CICS system C62D2FM5).

After the name of a resource is specified, the rest of the File Manager editor options work the same. For example, you can specify a template to format the data in the specified resource. An edit session for a CICS temporary storage queue with a template is shown in Figure 8-2.

```
  Process   Options   Help
 ------------------------------------------------------------------------------
 Edit             TS:C62D2FM5:RBDEMO                             Top of 40
 Command ===> ____________________________________________________ Scroll CSR
                                              Record AT TOP      Format TABL
        REC-TYPE NAME             EMPLOYEE-NO    AGE   SALARY     MONTH(1)
        #2       #3                        #4     #5       #6           #7
        AN 1:2   AN 3:20              BI 23:2 BI 25:2  PD 27:4      BI 31:4
        <>       <---+----1----+---->    <---+> <---+> <---+--> <---+----1>
 ****** ****  Top of data  ****
 000001 01       Grant Smith                23644     83    50001  1263225675
 000002 01       Andrew Apple                6645     53    78500          30
 - - -  - - - - - - - REC-TYPE02  - - - - - - - - - - 2 Line(s) suppressed
 000005 01       Ted Dexter                  3327     52    60250          14
 000006 01       Roddy Armstrong             5683     34    77000          28
 000007 01       Cliff Roberts               2265     57   100000          44
 000008 01       James Browne                1117     46   125000          47
 000009 01       Silvia Carrot               2308     29    61400           1
 000010 01       Dan Peters                  4479     54    63000          38
 000011 01       John Laws                   3422     21    46750          14
 000012 01       Liz Childs                  3439     55    66000          44
 000013 01       Bill McCork                 4565     40    76200          10
  F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
  F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 8-2 Edit session for CICS temporary storage queue with a template

When you are in an edit session, the File Manager editor behaves the same. For example, you can change SHOW and SHADOW settings to restrict the set of records for display, and issue various editor commands, such as **Find/Change**.

By using the **SAVEAS** command you can save a copy of the currently edited resource to a different resource. For example, if you are browsing a CICS resource or a WebSphere MQ queue and discovered something wrong in the live data, you can choose to save the data to your data set so that you can keep a copy of problematic data permanently for your own use. You might want to update your application to cope with the problematic data by using the saved data. You can specify different types of resource as a target resource of the **SAVEAS** command by using the resource identification convention that is described here.

8.1.2 Copying data by using File Manager copy utility function

By using the File Manager copy utility function (DSC), you can copy some data from a resource to another resource. You can specify any combination of data sets, UNIX System Services files, WebSphere MQ, and CICS resource as a TO/FROM target for the utility function. You can run the copy function online under ISPF or run it as a batch job.

When the File Manager copy utility function is started with a template, it gives you flexibility in terms of how you want to copy your data by using one of the following methods:

- Select fields to copy.
- Specify a scrambling option so that sensitive data is masked.

In the following example, we demonstrate how to copy a subset of data from a CICS temporary storage to a WebSphere MQ queue by using a template:

1. Prepare a template. Example 8-1 and Example 8-2 are the source code for creating templates, which describe data that is stored in the source and destination.

Example 8-1 Copybook for describing data source

```
01 People1.
   03 PersonID             PIC S999.
   03 FirstName            PIC A(10).
   03 Surname              PIC A(10).
   03 Phone                PIC X(14).
   03 Postcode             PIC X(5).
   03 Country              PIC X(44).
```

Example 8-2 Copybook for describing data destination

```
01 People2.
   03 PersonID             PIC S999.
   03 Surname              PIC A(10).
   03 Phone                PIC X(14).
   03 Country              PIC X(44).
```

2. Create templates by using File Manager's Template Workbench (option 7.1). Figure 8-3 shows the Template Workbench while a template is created from a copybook. The template 'KEEPER.TEMPLATE(PEOPLE1)' is created by using a copybook 'KEEPER.COPYBOOK.COBOL(PEOPLE1)'.

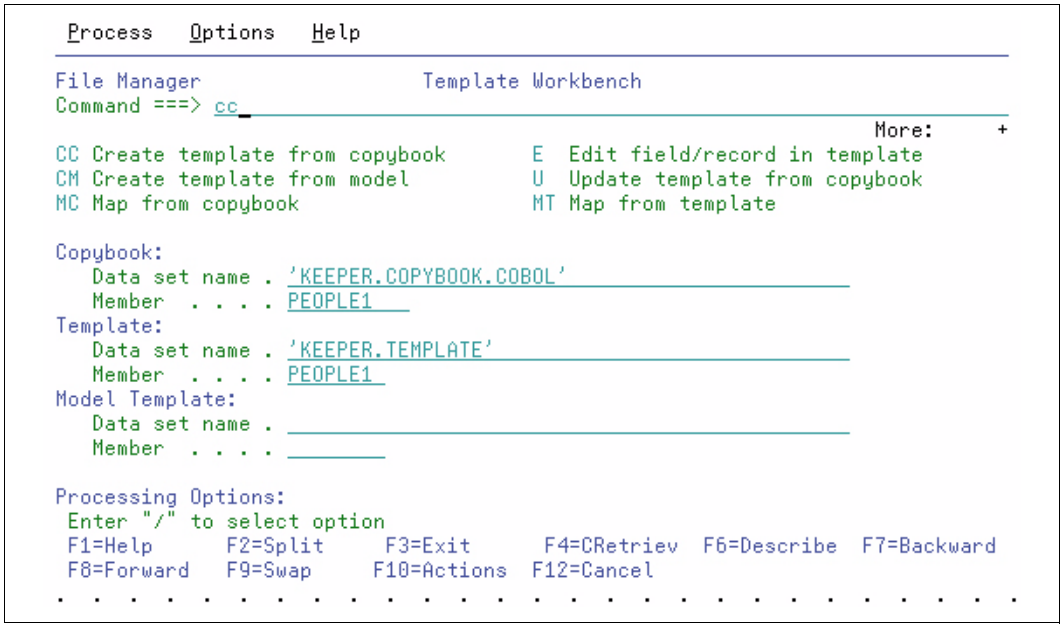


Figure 8-3 Template Workbench for creating template from copybook

3. Specify field mapping so that the subsets of fields are copied from the source to its destination. You start the field-mapping editor by completing the following steps:
   a. Specify the FROM template ('KEEPER.TEMPLATE(PEOPLE1)').
   b. Execute the **MT** command from the Template Workbench.
   c. Specify the TO template ('KEEPER.TEMPLATE(PEOPLE2)').
   d. Execute the **GE** command.

Figure 8-4 shows the Field-Mapping editor with the generated default mapping. Only a subset of fields is mapped. That is, only the mapped fields are copied when the File Manager copy function copies data by using those templates.

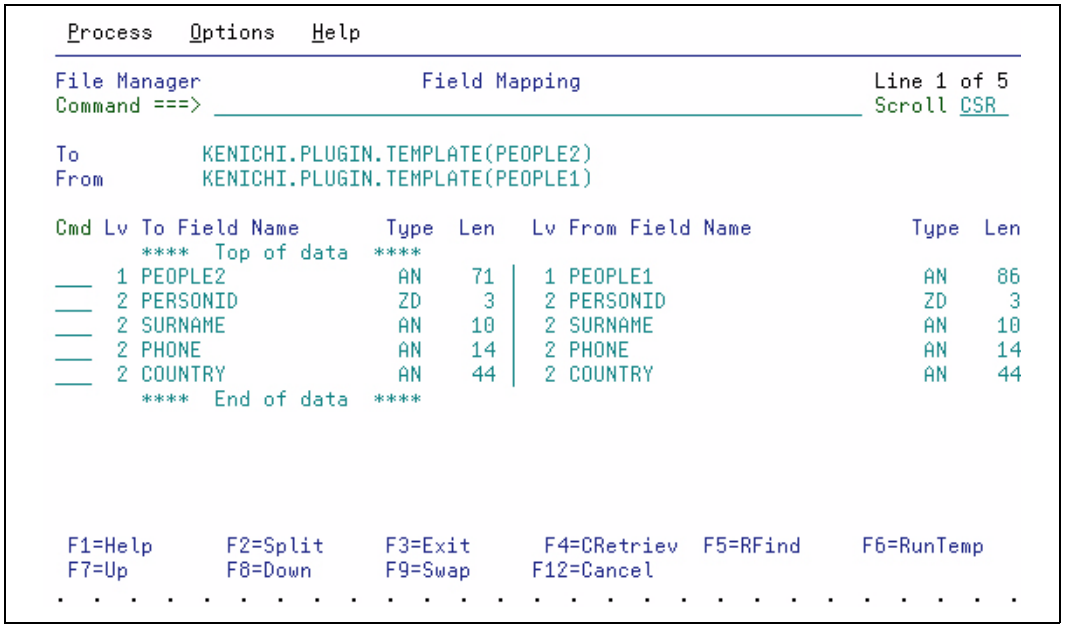


Figure 8-4 The default mapping that is generated by File Manager

To make the scenario more interesting, we use the following procedure to specify the scrambling option for the phone number field to protect the privacy of people if you are working with the real production data:

a. Specify “e” next to the PHONE field.
b. In the field attributes panel, specify the **1. Random** option under the scramble option.
c. Exit the panel.
d. Exit and save the template.

All field-mapping information is stored in TO template (‘KEEPER.TEMPLATE(PEOPLE2)’ in the example).

4. Open the File Manager copy function panel (option 3.3).

5. Specify the data source information and its template (see Figure 8-5).

```
 Process    Options    Help
──────────────────────────────────────────────────────────────────────────────
File Manager                      Copy Utility
Command ===> _________________________________________________________________
                                                                  More:     +
From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . TS:C62D2FM5:RBDEMO____________________________  +
   Member  . . . . . . . . ________    (Blank or pattern for member list)
   Volume serial . . . . . ______      (If not cataloged)
   Start key . . . . . . . ______________________________ key or slot
   Skip count  . . . . . . 0_______    Include _________  Repeat skip _________
   Copy count  . . . . . . ALL_____    number of records to be copied
From Copybook or Template:
   Data set name . . . . . 'KENICHI.PLUGIN.TEMPLATE(PEOPLE1)'____________
   Member  . . . . . . . . __________    (Blank or pattern for member list)
Processing Options:
   Copybook/template    Enter "/" to select option
   1  1. Above          _  Batch execution          _  Advanced member selection
      2. None           _  Use proc ________        _  Skip member name list
      3. Create dynamic _  Ignore length mismatch _  REXX member selection: P
   _  Edit template     _  JCL Source format        _  Directory integrity
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 8-5 Copy utility data source specification panel (FROM)

6. Specify the data destination information and its template (see Figure 8-6).

```
 Process    Options    Help
──────────────────────────────────────────────────────────────────────────────
Copy from TS:C62D2FM5:RBDEMO
Command ===> _________________________________________________________________
                                                                  More:     +
To Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . MQ:MQ72:KENICHI.TESTQ1________________________  +
   Member name (or mask) . ________      (Blank or pattern for member list)
   Volume serial . . . . . ______        (If not cataloged)

To Copybook/Template From:  KENICHI.PLUGIN.TEMPLATE(PEOPLE1)
   Data set name . . . . . 'KENICHI.PLUGIN.TEMPLATE(PEOPLE2)'____________
   Member  . . . . . . . . __________    (Blank or pattern for member list)

Processing Options:
  Copybook/template usage   Disposition          Enter "/" to select option
  1  1. Above               1  1. Old or Reuse   _  Replace members
     2. None                   2. Mod            _  Edit template mapping
     3. Create dynamic      Stats Option         _  Edit template source
  ISPF Packing              _  1. Off            _  Binary mode, reclen _____
  1  1. Asis                   2. Force
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 8-6 Copy utility data destination panel (TO)

7. Execute the copy utility function. If you select the **Batch execution** option, it generates a sample JCL. Otherwise, the File Manager copy function is executed online.

You can specify any combination of different resources when you are preparing to execute the File Manager copy utility function.

8.1.3 Comparing data by using File Manager compare utility function

In this section, we demonstrate how to compare two CICS resources in two CICS regions. You can always specify any combination of different resource types as you see fit for your task.

Complete the following steps to compare data by using the File Manager compare utility function:

1. Start the Compare Utility (option 3.11).
2. Specify OLD resource (see Figure 8-7).

```
 Process   Options   Help
 ------------------------------------------------------------------------
 File Manager          Compare Utility : Old Data Set
 Command ===> _                                                          
                                                                  More:  +
 "Old" Partitioned, Sequential or VSAM Data Set, or HFS file:
    Data set/path name  . . TS:C62D2FM5:RBDEMO                          +
    Member  . . . . . . . . ________    (Blank or pattern for member list)
    Volume serial . . . . . ______      (If not cataloged)
    Start key . . . . . . . ____________________________ key or slot
    Skip count  . . . . . . 0_______    number of records to be skipped
    Compare count . . . . . ALL_____    number of records to be compared

 "Old" Copybook or Template:
    Data set name . . . . . 'KENICHI.PLUGIN.TEMPLATE'
    Member  . . . . . . . . PEOPLE___   (Blank or pattern for member list)

 Processing Options:
    Copybook/template usage           Enter "/" to select option
    1  1. Above                       _  Edit template  _ Type (1,2,S)
       2. Previous                    _  Advanced member selection
       3. None                        _  Skip member name list
  F1=Help      F2=Split     F3=Exit     F4=Expand    F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 8-7 Compare utility: OLD resource specification panel

3. Specify NEW resource (see Figure 8-8).

```
 Process   Options   Help
 ------------------------------------------------------------------------
 File Manager          Compare Utility : New Data Set
 Command ===> _                                                          
                                                                  More:  +
 "New" Partitioned, Sequential or VSAM Data Set, or HFS file:
    Data set/path name  . . TS:C63D2FM1:RBDEMO                          +
    Member  . . . . . . . . ________    (Blank or pattern for member list)
    Volume serial . . . . . ______      (If not cataloged)
    Start key . . . . . . . ____________________________ key or slot
    Skip count  . . . . . . 0_______    number of records to be skipped
    Compare count . . . . . ALL_____    number of records to be compared

 "New" Copybook or Template:
    Data set name . . . . . 'KENICHI.PLUGIN.TEMPLATE'
    Member  . . . . . . . . PEOPLE___   (Blank or pattern for member list)

 Processing Options:
    Copybook/template usage           Enter "/" to select option
    1  1. Above                       _  Edit template  _ Type (1,2,S)
       2. Previous                    _  Binary mode, reclen _____
       3. None
  F1=Help      F2=Split     F3=Exit     F4=Expand    F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 8-8 Compare utility - NEW resource specification panel

4. Specify various compare options.

Execute online or batch.

The key point here is that you can compare two CICS resources from different CICS regions by using the File Manager compare utility function.

8.2 Data manipulation by using File Manager plug-in for Eclipse

In this section, we describe various types of data manipulation by using File Manager.

8.2.1 Edit or browse resources by using File Manager editor

Figure 8-9 shows the Systems Information view, which is provided by the File Manager plug-in for Eclipse. As you can see, all of the resources that are available on your system are represented as a tree in the view[2].

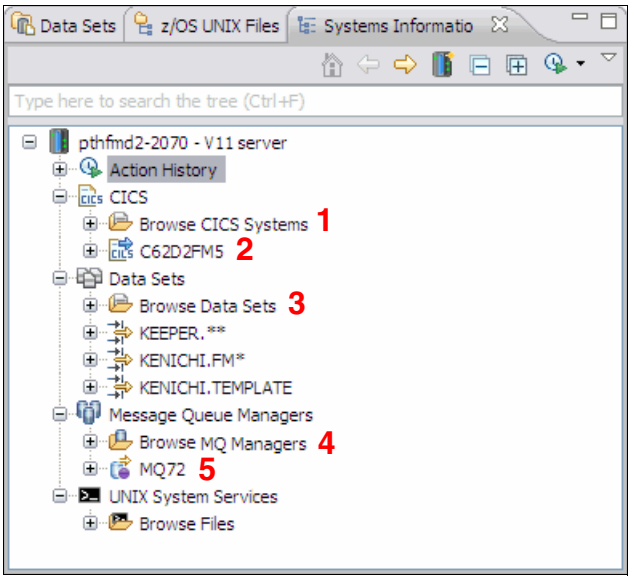


Figure 8-9 File Manager Systems Information presents all accessible resources in one view

All File Manager actions, such as the following edit and utility functions, are available from this view that is based on the current selection:

1. Browse CICS Systems: If you expand this node, you see a list of CICS Systems that are known to the File Manager server[3]. By using this ability you can browse to a different CICS resource.
2. Query a CICS resource: You can create a specific query to look up a specific CICS resource (such as CICS files that start with the name A: “FI:applid:A*”). Such queries are grouped by the CICS System.
3. Browse Data Sets: If you expand this node, you can see all of the data sets with your user name as their high-level qualifier.
4. Browse WebSphere MQ Manager: If you expand this node, you see all of the WebSphere MQ queue managers that are known to File Manager server.
5. Query queues: You also can create a specific query to look up queues for a queue manager. Such queries are grouped by their queue manager name.

[2] You can make a connection to multiple systems from the Systems Information view.
[3] You must customize and allocate FMNCICS DD in your server job.

To start an edit/browse session, you must browse to a resource in the view and start the File Manager editor from the context menu. Complete the following steps to start the session:

1. Create a connection to your z/OS system and right-click the CICS node.
2. Select **Add query from the context menu**.
3. In the Add CICS resource query window, specify the CICS application ID, resource type, and resource name pattern (as shown in Figure 8-10). You can type the application ID of your CICS system or click **Lookup** to look up CICS systems. In this example, we are looking up all temporary storage queues that are defined to a CICS system.

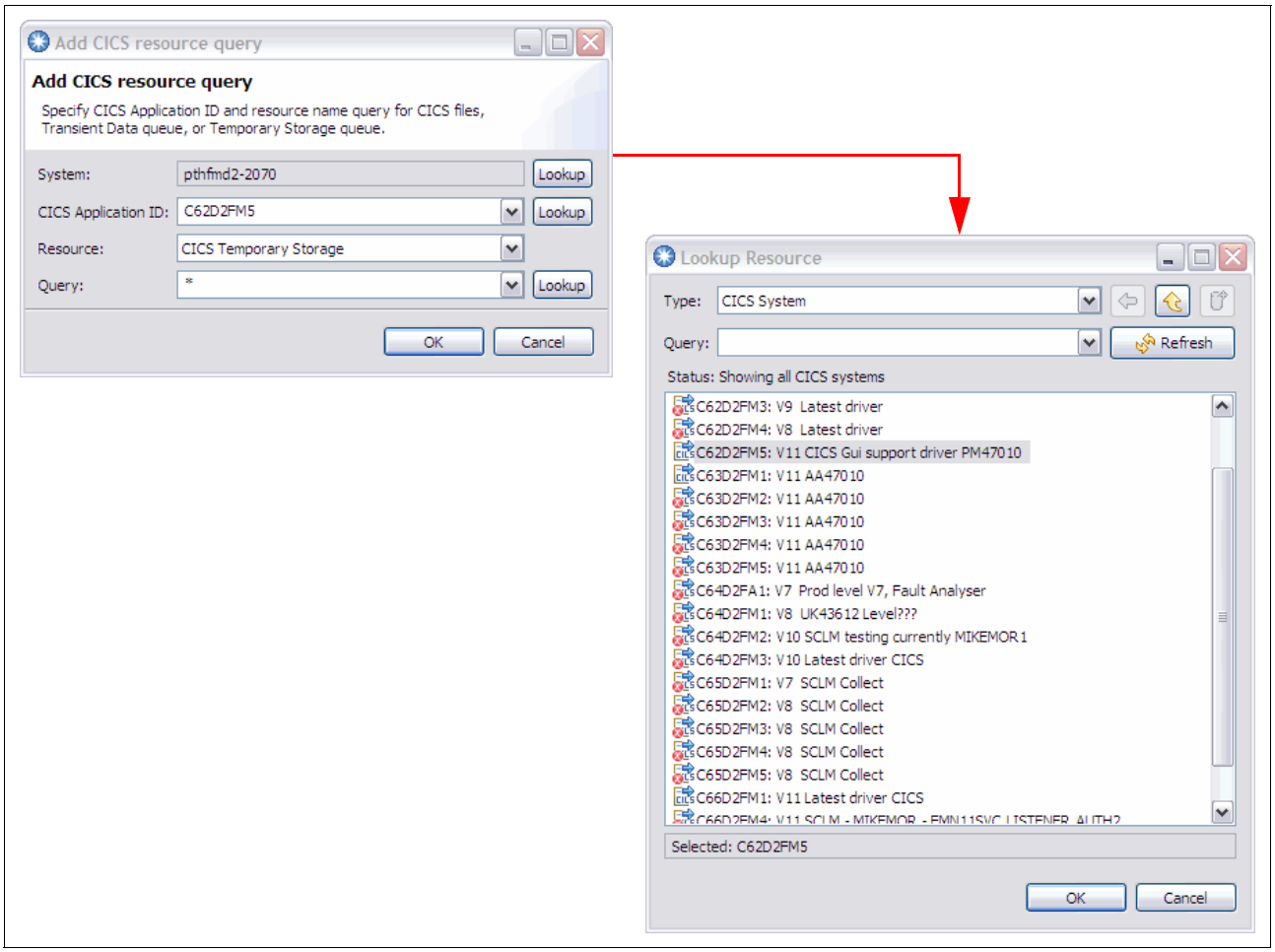


Figure 8-10 Lookup Resource dialog for looking up the CICS system name

4. Select **OK** and expand the query node in the Systems Information view.

5. Select a temporary storage queue and open the context menu. Select **Formatted Editor** from the context menu, as shown in Figure 8-11.

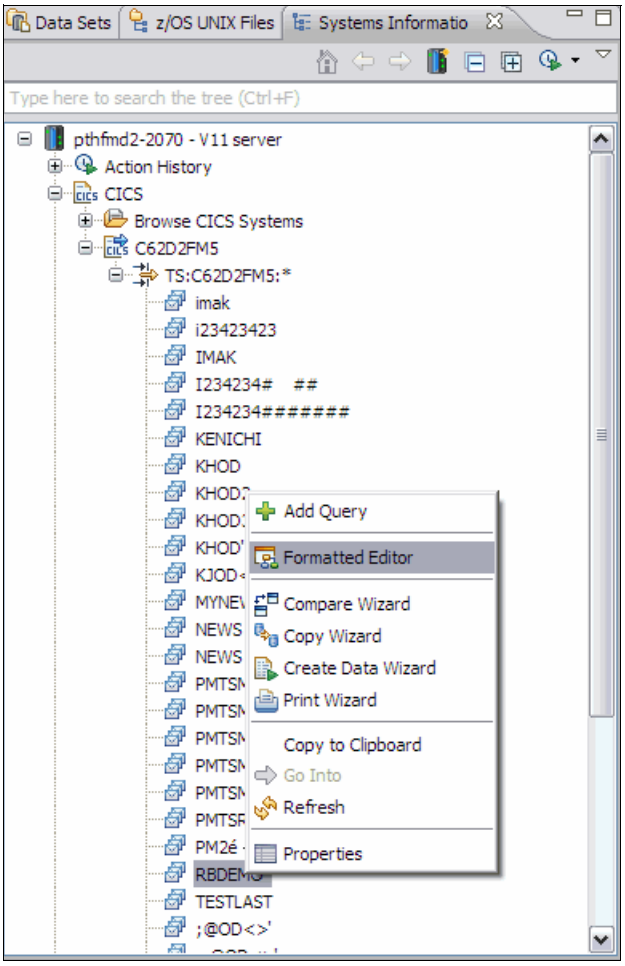


Figure 8-11 Invocation of File Manager editor from Systems Information view

6. Specify a File Manager editor option (such as specifying a template) and click **Edit** or **View** to start the editor session for the selected resource (as shown in Figure 8-12).

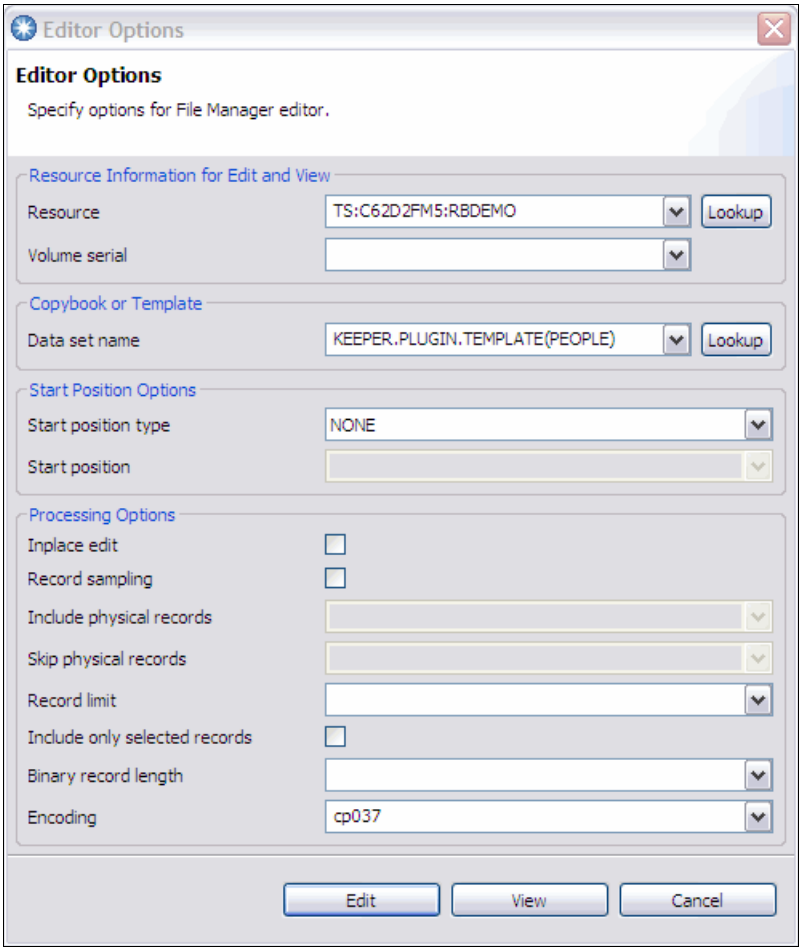


Figure 8-12 Editor options window

7. When you are in an edit session (as show in Figure 8-13), all actions behave the same regardless of which resource is being edited.

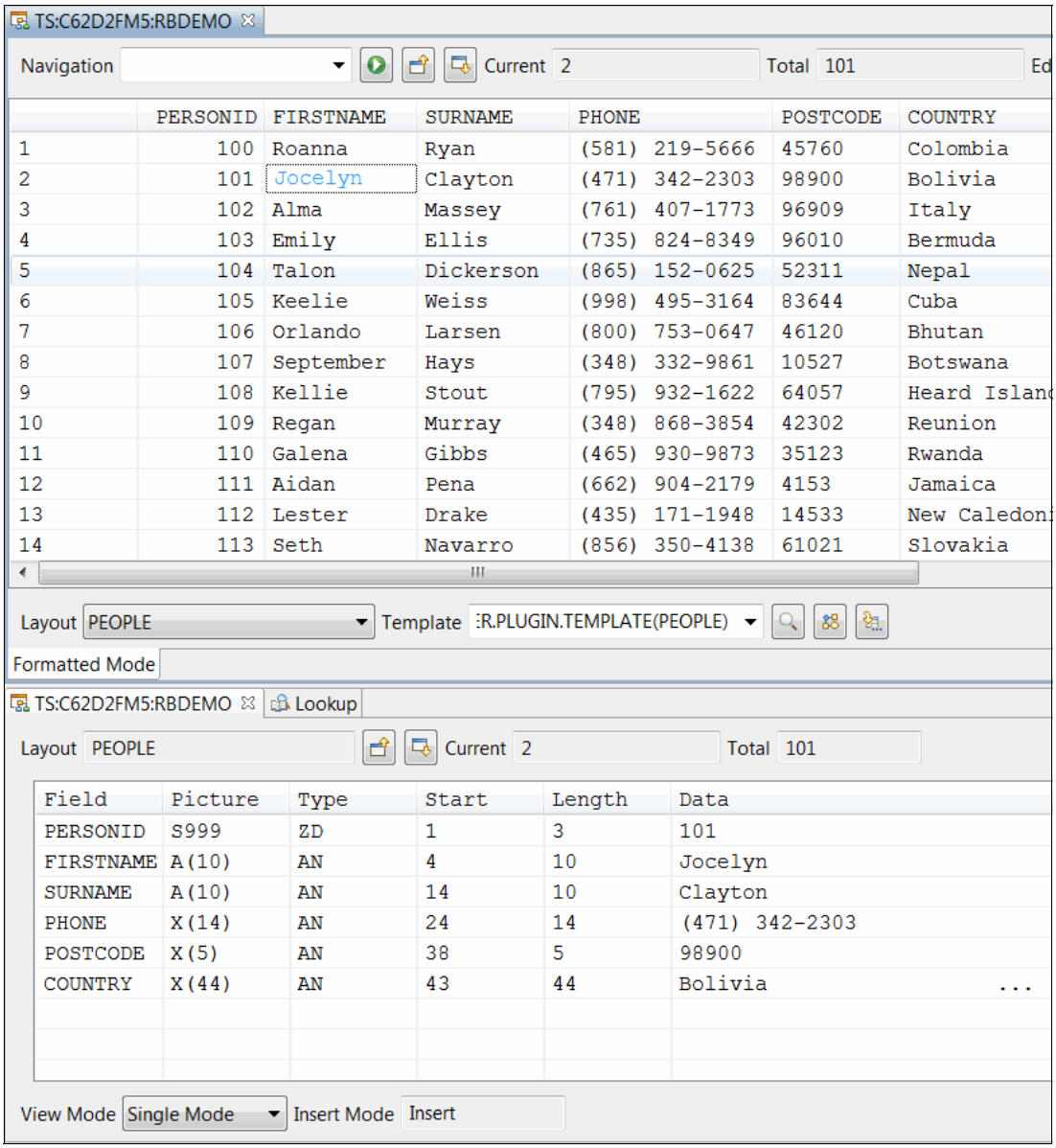


Figure 8-13 Editor session of CICS temporary storage

Alternatively, you can change the resource to edit dynamically while you are in the editor's options window (as shown in Figure 8-12 on page 275). The Resource name is filled in with the name of resource that was selected in the Systems Information view when you started the editor. You can specify a different resource by using one of the following options:

- If you know the name of the resource that you want to edit, then overwrite the resource name. The same naming conventions apply as described in 8.1.1, "Edit or browse resources by using File Manager editor" on page 264.
- You can click **Lookup**, which brings up the Lookup Resource window (as shown in Figure 8-10 on page 273). By using the window, you can interactively select the target resource.
- You can enter the name of resource while you are using the content assist[4]. You can start the content assist by pressing Ctrl+Spacebar while you are in the resource name field. Depending on what you specified in the field, the content assist looks up the resources that are available on your system. The following examples can be used:
  - Enter nothing and press Ctrl+Spacebar: Looks up all of the HLQ that is available on your system.
  - Enter TS: and press Ctrl+Spacebar: Looks up all of the CICS systems.
  - Enter TS:C62D2FM5:* and press Ctrl+Spacebar: Looks up all temporary storage queues that are available on a CICS system called C62D2FM5.
  - Enter MQ: and press Ctrl+Spacebar: Looks up all of the WebSphere MQ queue managers that are on your system.

SAVE AS action is available, with which you can save data in the current edit session to another resource. You can choose to save data from any resource type to any resource type[5].

[4] Content assist for looking up a resource is provided throughout the File Manager plug-in, where applicable.
[5] You cannot choose a CICS file as a target for SAVE AS action.

8.2.2 Copying data by using the File Manager copy utility function

In the following scenario, we walk through the same copy data scenario that was described in 8.1.2, “Copying data by using File Manager copy utility function” on page 266. However, we now use the Eclipse plug-in:

1. Create a query to look up your copybooks and locate the copybooks in the Systems Information view.
2. Select two copybooks (PEOPLE1 and PEOPLE2) and select **Template Editor** from the context menu, as shown in Figure 8-14. This method creates a template dynamically from the selected copybook.

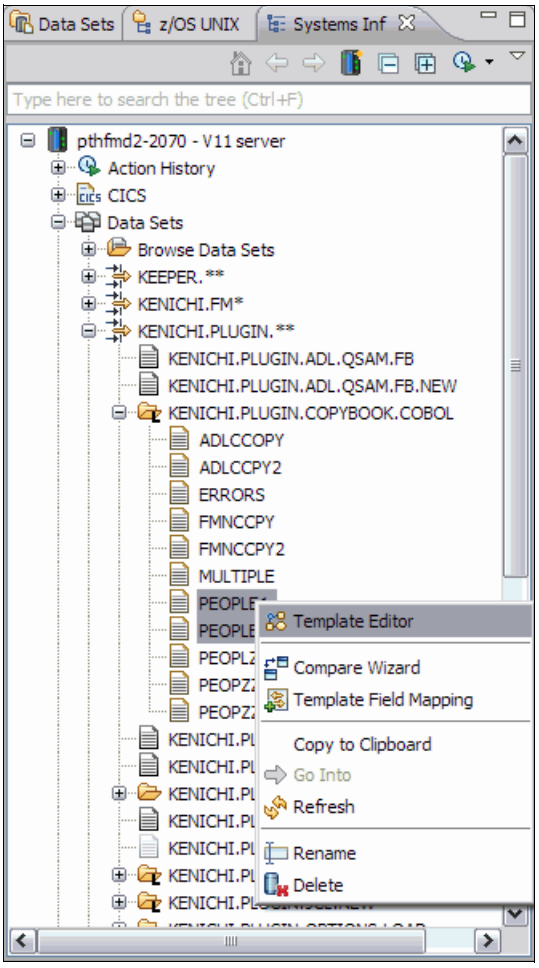


Figure 8-14 Invocation of Template Editor from Systems Information view

3. When you see two instances of Template Editor opened in your workspace, save them to a suitable data set (such as KEEPER.TEMPLATE). You must choose the **SAVE AS** option so that you can use them as a template and change later.

4. Create a query and select the PEOPLE2 template that you created in the previous step.
5. Select **Template Field Mapping** from the context menu. The Template Field Mapping Editor window appears, as shown in Figure 8-15.

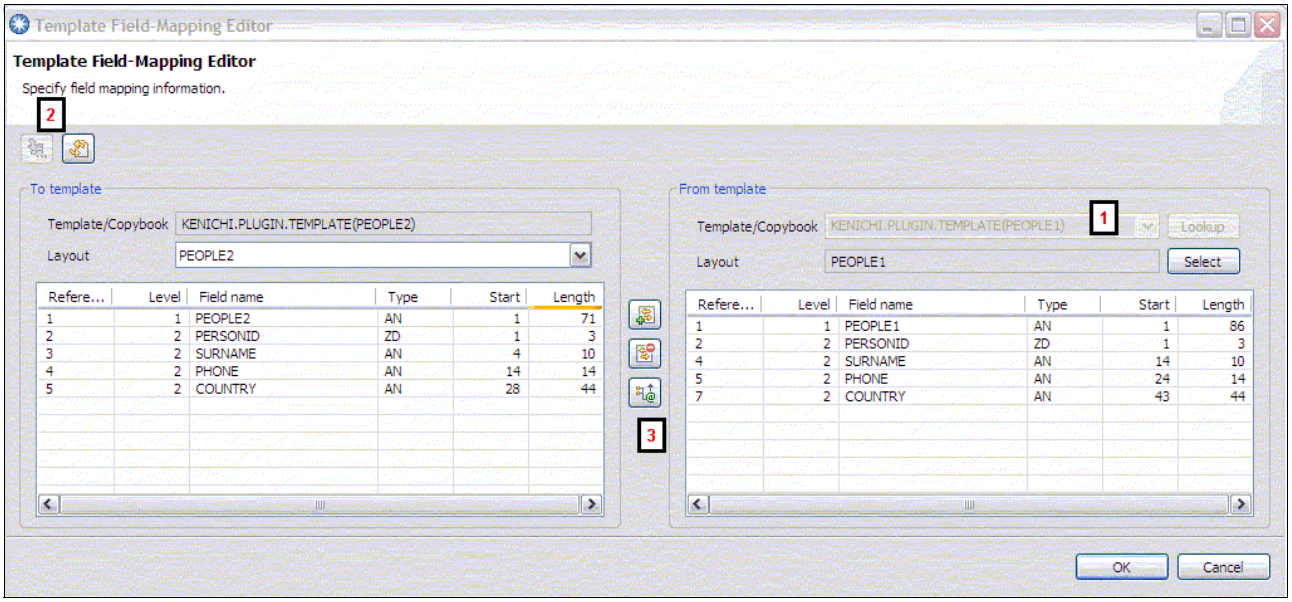


Figure 8-15 Template field-mapping editor

6. The TO and FROM templates are pre-populated with PEOPLE2 template. Change the FROM template to PEOPLE1[6] (Number 1 in Figure 8-15).
7. Click **Load** (in the upper left of the window) to start the field-mapping session (Number 2 in Figure 8-15).
8. Select a row for the Phone field. Click **Attributes** (Number 3 in Figure 8-15) to open the window for specifying attributes for the selected field.

[6] You can select two templates and start the field-mapping editor. The first selection is used as the TO template. The second selection is used as the FROM template. In this scenario, the order is wrong, hence we selected only the PEOPLE2 template.

9. Select **Random option** and close the attribute, as shown in Figure 8-16 on page 280.

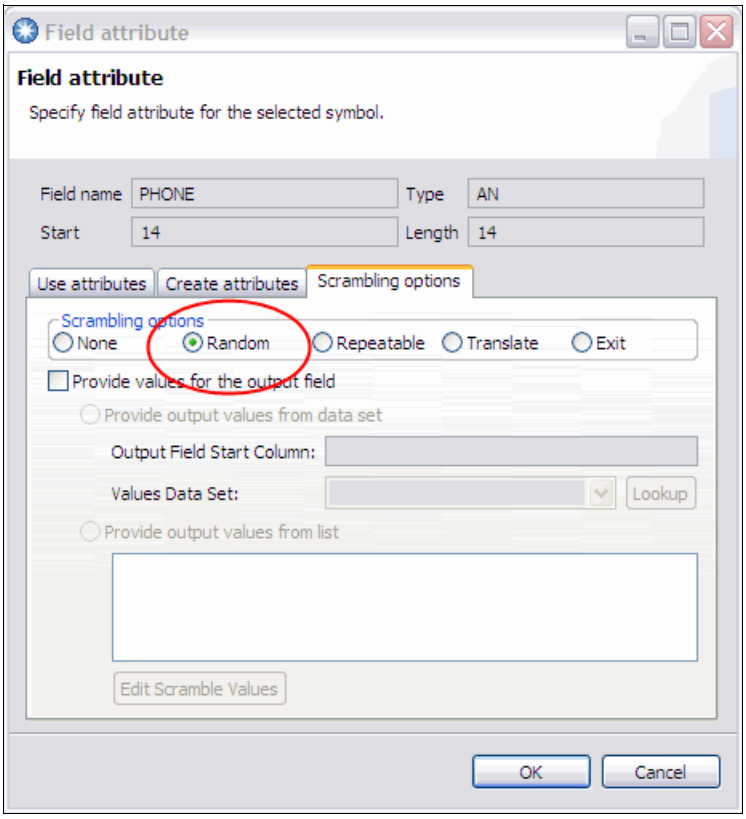


Figure 8-16 Field attribute dialog for specifying attributes for a field in a template

10. Close the field-mapping editor window.
11. Create a query for the CICS resource that you want to copy from (for example, TS:C62D2FM5:RB*). Expand the query node and select the resource to copy from.
12. Right-click the resource and select **Copy Wizard** from the context menu. You see the Copy Wizard. The selected resource must be selected as the source.

13. Specify the copy destination (such as MQ:MQ72:KENICHI.TESTQ1), as shown in Figure 8-17.

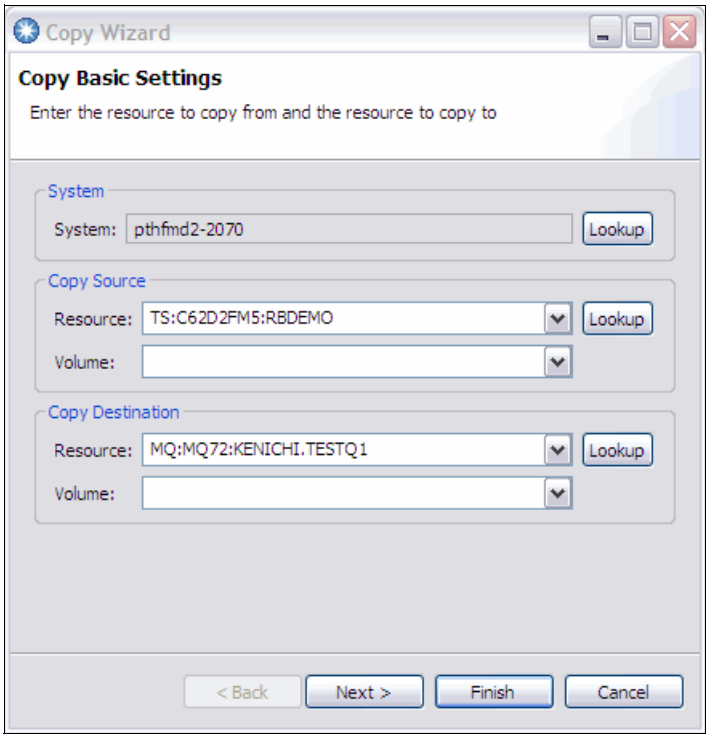


Figure 8-17 Copy wizard resource specification page

14. Specify the templates to use for the copy function, as shown in Figure 8-18.

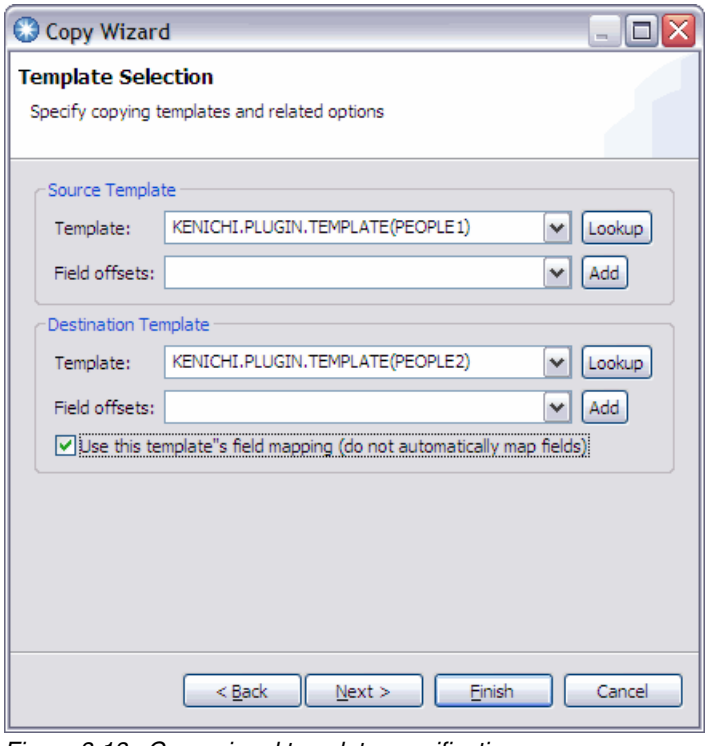


Figure 8-18 Copy wizard template specification page

15. Follow the rest of wizard to specify other File Manager copy utility function parameters.

    Now the data from the selected CICS TS queue must be copied to the specified WebSphere MQ queue.

8.2.3 Comparing data by using File Manager compare utility function

In this section, we demonstrate how to start File Manager's compare utility function to compare two CICS TS queues from different CICS systems as described in 8.1.3, "Comparing data by using File Manager compare utility function" on page 270. Complete the following steps to specify different sets of resources to compare:

1. Create a query to look up a TS queue from a CICS region (such as TS:C62D2FM5:RB* for looking up TS queues with a name that starts with RB at CICS system called C62D2FM5).
2. Create another query to look up a TS queue from a different CICS region (such as TS:C63D2FM1:RB* for looking up TS queues with a name that starts with RB at CICS system called C63D2FM1).
3. Select two TS queues to compare in the Systems Information view, as shown in Figure 8-19.

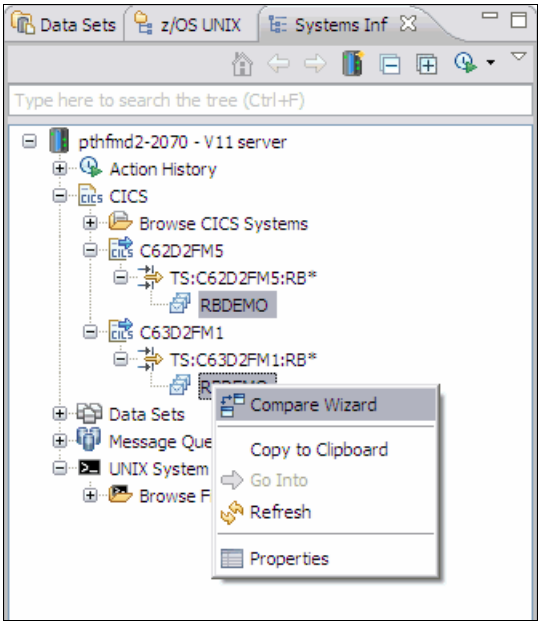


Figure 8-19 Starting the compare wizard against two CICS TS queues from different CICS systems

4. Select **Compare Wizard** from the context menu.
5. The selected resources must be used as NEW and OLD resources to compare in the wizard. Follow the wizard to specify various File Manager compare utility function parameters.
6. Click **Finish** to start the compare utility function. The result is displayed in your local text editor.

9

Using IBM PD Tools with Java

In this chapter, we describe how to use Fault Analyzer and Debug Tool with Java on z/OS.

9.1 Analyzing Java dumps with Fault Analyzer

In this section, we describe how to use Fault Analyzer to analyze dumps from a Java virtual machine. In the chapter “Real-time analysis” of the *IBM Fault Analyzer for z/OS User’s Guide and Reference,* SC19-3131-05, two methods are described for obtaining a Java dump. For this Redbooks publication, we use the -Xdump option that is available to IBM Java environments.

Consider the Java program that is shown in Figure 9-1.

```
public class RedbookGCD
{
       public static void main( String [] args )
       {
              System.out.println( "GCD of " + args[0] + " and " + args[1] );
              new RedbookGCD().gcd( Long.parseLong( args[0] ), Long.parseLong( args[1] )
);
       }

       private void gcd( long _one, long _two )
       {
              System.out.println( "Recurse: " + _one + " " + _two );
              if( _one > _two )
              {
                     this.gcd( _two, _one % _two );
              }
              else
              {
                     this.gcd( _one, _two % _one );
              }
       }
}
```

Figure 9-1 RedbookBatch.java

The main method takes two arguments as Integers and calls the Global Catalog Data Set (GCD) method. The `gcd` method recursively calls itself. Eventually, one of the arguments reaches 0, which is not treated as the final state of the recursive definition. An exception is thrown, as shown in Figure 9-2.

```
$ java RedbookGCD 9832984798732984 12321453153
GCD of 9832984798732984 and 12321453153
Recurse: 9832984798732984 12321453153
Recurse: 12321453153 9288872323
Recurse: 9288872323 3032580830
Recurse: 3032580830 191129833
Recurse: 191129833 165633335
Recurse: 165633335 25496498
Recurse: 25496498 12654347
Recurse: 12654347 187804
Recurse: 187804 71479
Recurse: 71479 44846
Recurse: 44846 26633
Recurse: 26633 18213
Recurse: 18213 8420
Recurse: 8420 1373
Recurse: 1373 182
Recurse: 182 99
Recurse: 99 83
Recurse: 83 16
Recurse: 16 3
Recurse: 3 1
Recurse: 1 0
Exception in thread "main" java.lang.ArithmeticException: divide by zero
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.gcd(RedbookGCD.java:15)
        at RedbookGCD.main(RedbookGCD.java:7)
```

Figure 9-2 Recursive GCD function not finishing upon reaching 0

To capture a dump by using the Xdump option, we must understand which event we want a dump to be created for. In this case, we want to capture the dump when the exception is thrown. The next question is, what type of exception do we want to capture a dump on the event of a throw; it is `java.lang.ArithmeticException`.

The modified command line to run the program is shown in Figure 9-3.

```
java -Xdump:system:events=throw,filter=*Arithmetic* RedbookGCD 9832984798732984
12321453153
```

Figure 9-3 Command line

This command line results in a dump in data set `KYKWAN.JVM.TDUMP.KYKWAN5.D111018.T070510`. You can supply options for Java to create a custom name, and to immediately run a program on that dump. The chapter “Using dump agents” of the manual *Java Diagnostics Guide*, explains what the supported events are and specific options for each event. The guide is available at this website:

http://publib.boulder.ibm.com/infocenter/javasdk/v5r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.50%2Fdiag%2Fwelcome.html

The *IBM Fault Analyzer for z/OS User's Guide and Reference,* SC19-3131-05, explains how to configure Java to call Fault Analyzer on the generated dump.

Analyzing a Java dump is the same as with any normal interactive dump reanalysis. When a Java dump is analyzed, a DTFJ process is started in parallel (as shown in Figure 9-4), which might not be completed for a few minutes.

```
Confirm Java Fault Entry Reanalysis                          Line 1 Col 1 76
Command ===>                                               Scroll ===> CSR

Java DTFJ processing has not yet completed for this fault entry.

Press Enter to continue reanalysis with incomplete Java information, or
press PF3/PF12 to cancel.

*** Bottom of data.









 F1=Help     F3=Exit     F5=RptFind  F7=Up       F8=Down     F12=Cancel
```

Figure 9-4 Wait for DTFJ processing to finish

When DTFJ processing finishes, the main interactive reanalysis panel shows the new option Java Information, as shown in Figure 9-5.

```
 File  View  Services  Help

Interactive Reanalysis Report                                                    Line 1 Col 1 132
 Command ===>                                                                     Scroll ===> CSR
 JOBNAME: KYKWAN5   ABEND: n/a                  FAE1      2011/10/18  15:05:10

 Fault Summary:
 Module /apc/java531-UK53749/usr/lpp/java/J5.0/bin/libj9prt23.so offset X'716':
 No abend could be determined--possible loop or wait situation.

 Select one of the following options to access further fault information:
   1. Synopsis
   2. Event Summary
   3. Java Information
   4. Storage Areas
   5. Language Environment Heap Analysis
   6. Abend Job Information
   7. Fault Analyzer Options

 *** Bottom of data.

  F1=Help     F3=Exit     F4=Dsect    F5=RptFind  F6=Actions  F7=Up       F8=Down    F10=Left    F11=Right
   *FAULTA
```

Figure 9-5 Updated main panel with Java information option

As shown in Figure 9-6, the Java Information panel shows the state of the JVM at the time of the dump, including environment variables, the JVM command-line argument, and threads that are active at the time of the dump.

```
 File  View  Services  Help

Java Information                                                                                    Line 32 Col 1 132
 Command ===>                                                                                        Scroll ===> CSR
 JOBNAME: KYKWAN5   ABEND: n/a                  FAE1      2011/10/18  15:05:10
 Java VM init args

   args

   arg=-Xjcl:jclscar_23
   arg=-Dcom.ibm.oti.vm.bootstrap.library.path=/apc/java531-UK53749/usr/lpp/java/J5.0/bin
   arg=-Dsun.boot.library.path=/apc/java531-UK53749/usr/lpp/java/J5.0/bin
   arg=-Djava.library.path=/apc/java531-UK53749/usr/lpp/java/J5.0/bin:/apc/java531-UK53749/usr/lpp/java/J5.0/bin/classic/libjvm.so:
   /apc/java531-UK53749/usr/lpp/java/J5.0/bin/classic:/apc/java531-UK53749/usr/lpp/java/J5.0/bin/:/u/kykwan/java_test:/lib:/usr/lib
   :/apc/java531-UK53749/usr/lpp/java/J5.0/bin:/apc/java531-UK53749/usr/lpp/java/J5.0/bin/classic:/u/kykwan/idijlib:.:
   arg=-Djava.home=/apc/java531-UK53749/usr/lpp/java/J5.0
   arg=-Djava.ext.dirs=/apc/java531-UK53749/usr/lpp/java/J5.0/lib/ext
   arg=-Duser.dir=/u/kykwan/java_test
   arg=_j2se_j9=70912
   arg=-Xdump
   arg=vfprintf
   arg=-Xdump:system:events=throw,filter=*Arithmetic*
   arg=-Dinvokedviajava
   arg=-Djava.class.path=/u/kykwan/java_test:/apc/java531-UK53749/usr/lpp/java/J5.0/lib/tools.jar:/u/kykwan/idijlib/IDIServer.jar:/
   u/kykwan/idijlib/log4j-1.2.14.jar:
   arg=vfprintf
   arg=_port_library

 Java threads with traceback information

 Call trace for thread: main

 Method          Location
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
 RedbookGCD.gcd  RedbookGCD.java:15
  F1=Help     F3=Exit     F4=Dsect    F5=RptFind  F6=Actions  F7=Up       F8=Down    F10=Left    F11=Right
   *FAULTA
```

Figure 9-6 Java information panel that shows JVM arguments and thread call trace

As shown in Figure 9-7, the Event Summary panel is reworked to show Java events in place of established events that are generated by the operation of the JVM run time. The events work as normal.

```
 File  View  Services  Help

Event Summary                                                                                          Top of data
 Command ===>                                                                                        Scroll ===> CSR
 JOBNAME: KYKWAN5   ABEND: n/a                  FAE1      2011/10/18  15:05:10

 Event           Fail  Module   Program  EP
 #  Type         Point Name     Name     Name     Event Location (*) Description
  1 Call               CEEPLPKA n/a      CEEBBEXT E+1B8              BOOTSTRAP MODULE FOR Language Environment; From LPA
  2 Call               CELHV003 n/a      EDCZHINV E+B4               CRTL Main invocation event XPLINK; From not determined
  3 >>> XPLink         CEEPLPKA n/a      n/a      M+1CBD44           CEL Common Runtime; From LPA
  4 Call               java     main     main     E+2AF4             From /apc/java531-UK53749/usr/lpp/java/J5.0/bin/
  5 Java               n/a      n/a      RedbookGCD.main
                                                  L#7                From not determined
  6 Call               libj9vm23.so
                                triggerExceptionThrowEvent
                                         triggerExceptionThrowEvent
                                         E+2C               From /apc/java531-UK53749/usr/lpp/java/J5.0/bin/
```

Figure 9-7 Showing the Java event

Java source, if available according to the requirements described in the *IBM Fault Analyzer for z/OS User's Guide and Reference,* SC19-3131-05, are displayed in the event's details panel (as shown in Figure 9-8) and is reachable through the source line hotspot (as shown in Figure 9-9 on page 290).

```
 File  View  Services  Help

Event 5 of 18: Java                                                                    Line 1 Col 1 132
 Command ===>                                                                           Scroll ===> CSR
 JOBNAME: KYKWAN5   ABEND: n/a                  FAE1      2011/10/18  15:05:10

 Previous Event Details


 This event occurred in Class RedbookGCD Method main.

 Java source from /u/kykwan/java_test/RedbookGCD.java:
   Source
   Line #
       -5 public class RedbookGCD
       -4 {
       -3 .public static void main( String [] args )
       -2 .{
       -1 ..System.out.println( "GCD of " + args[0] + " and " + args[1] );
   000007 ..new RedbookGCD().gcd( Long.parseLong( args[0] ), Long.parseLong( args[1] ) );
       +1 .}
       +2
       +3 .private void gcd( long _one, long _two )
       +4 .{
       +5 ..System.out.println( "Recurse: " + _one + " " + _two );


 The class static variable information is not available.
 The object instance variable information is not available.


 Next Event Details

 *** Bottom of data.









  F1=Help     F3=Exit     F4=Dsect    F5=RptFind  F6=Actions  F7=Up       F8=Down    F10=Left    F11=Right
   *FAULTA
```

Figure 9-8 The Java source that surrounds the Java event

```
 File  View  Services  Help

RedbookGCD.main Java Source                                                                    Line 1 Col 1 132
 Command ===>                                                                                   Scroll ===> CSR
 JOBNAME: KYKWAN5   ABEND: n/a                   FAE1      2011/10/18  15:05:10

 Java source from /u/kykwan/java_test/RedbookGCD.java.

 Source
 Line #
 000001
 000002 public class RedbookGCD
 000003 {
 000004 .public static void main( String [] args )
 000005 .{
 000006 ..System.out.println( "GCD of " + args[0] + " and " + args[1] );
 000007 ..new RedbookGCD().gcd( Long.parseLong( args[0] ), Long.parseLong( args[1] ) );
 000008 .}
 000009
 000010 .private void gcd( long _one, long _two )
 000011 .{
 000012 ..System.out.println( "Recurse: " + _one + " " + _two );
 000013 ..if( _one > _two )
 000014 ..{
 000015 ...this.gcd( _two, _one % _two );
 000016 ..}
 000017 ..else
 000018 ..{
 000019 ...this.gcd( _one, _two % _one );
 000020 ..}
 000021 .}
 000022 }

 *** Bottom of data.




  F1=Help     F3=Exit     F4=Dsect    F5=RptFind  F6=Actions  F7=Up       F8=Down    F10=Left    F11=Right
   *FAULTA
```

Figure 9-9 The complete Java source

9.2 Debug Tool for Java calling COBOL programs

In this chapter, we describe how Debug Tool helps in end-to-end debugging of Java calling COBOL programs. The reader is supposed to be aware of some of the technologies that we mention about to enable the debugging of Java and COBOL programs. You should be familiar with the following technologies:

- JZOS: IBM toolkit library that facilitates the creation of a JVM instance to run Java applications in a batch environment. The Java applications also can access z/OS file systems, for example, MVS data sets that ordinary Java APIs do not support.
- Debug Tool's Delay Debugging feature: For more information, see Figure 1-1 on page 3.

The application architecture consists of a Java application that submits a stock sale order request and the COBOL program that contains business logic to execute the stock order that is based on a fixed stock commission. The Java and COBOL applications are on z/OS.

9.2.1 Rational Developer for System z setup

The Java artifacts that are shown Figure 9-10 are the Java source files from a Java project in IBM Rational Developer for System z. The `TradeWrap.java` is the wrapper code that is generated when you compile the COBOL program on the host (see Figure 9-21).

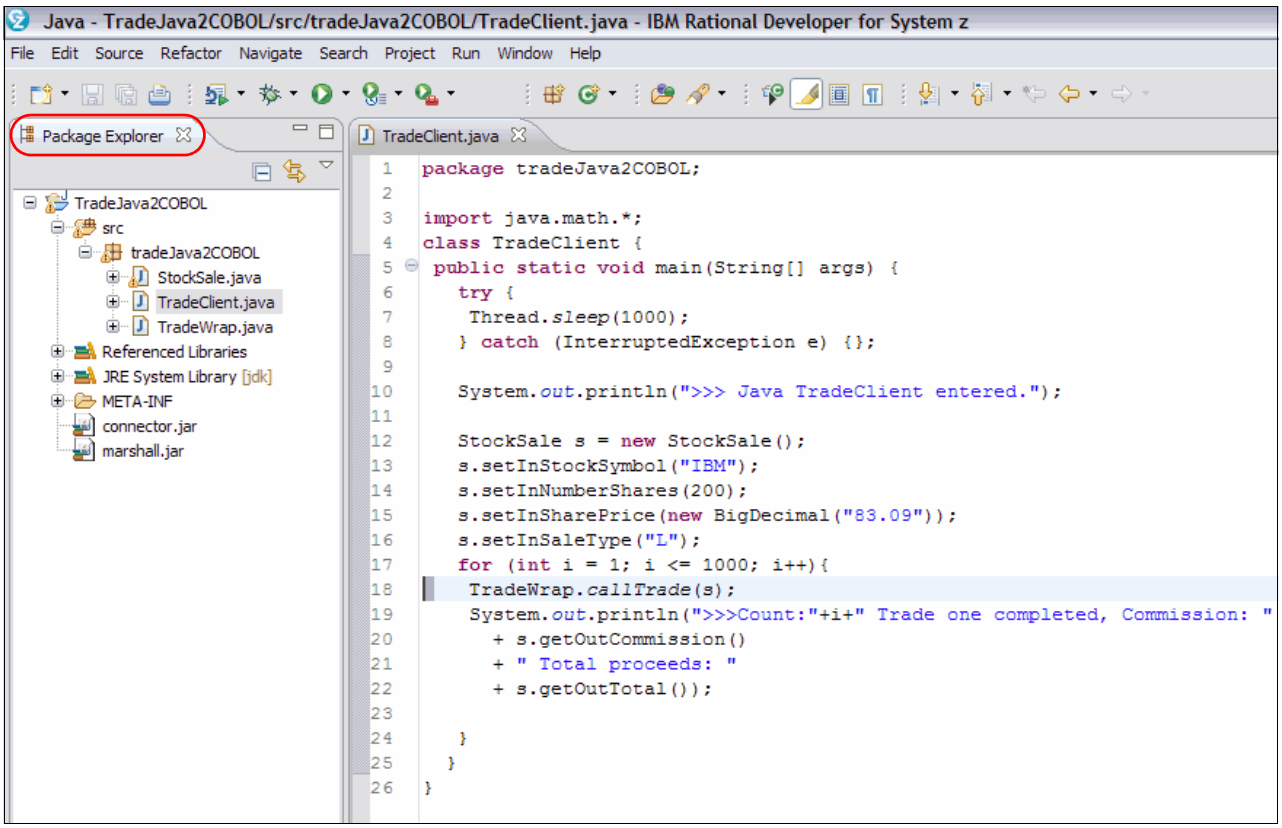


Figure 9-10 TradeJava 2COBOL project in the Package Explorer view

We define a Remote Java Application debug configuration in Rational Developer for System z. As shown in Figure 9-12 on page 293, use the Debug Configurations to create a new Remote Java Application configuration that is called TradeJava2COBOL and enter the host name and port number.

In Figure 9-12, the host is tlba07me.torolab.ibm.com with port 8010. The port number should match the value that is specified in the JZOS JCL that we review later in this chapter.

Complete the following steps to set up Rational Developer for System z:

1. In the Java perspective, click **Run** → **Debug Configurations** as shown in Figure 9-11.

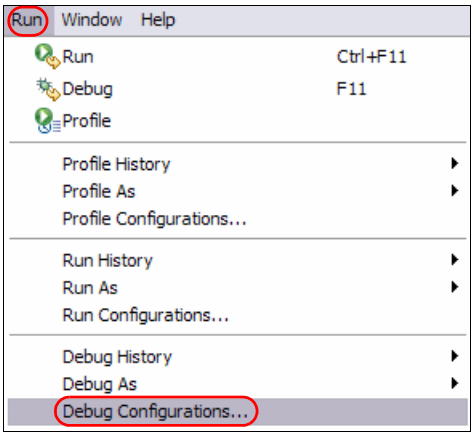


Figure 9-11 Run window

2. Create a Remote Java Application Debug configuration. Specify the host and port as shown Figure 9-12.

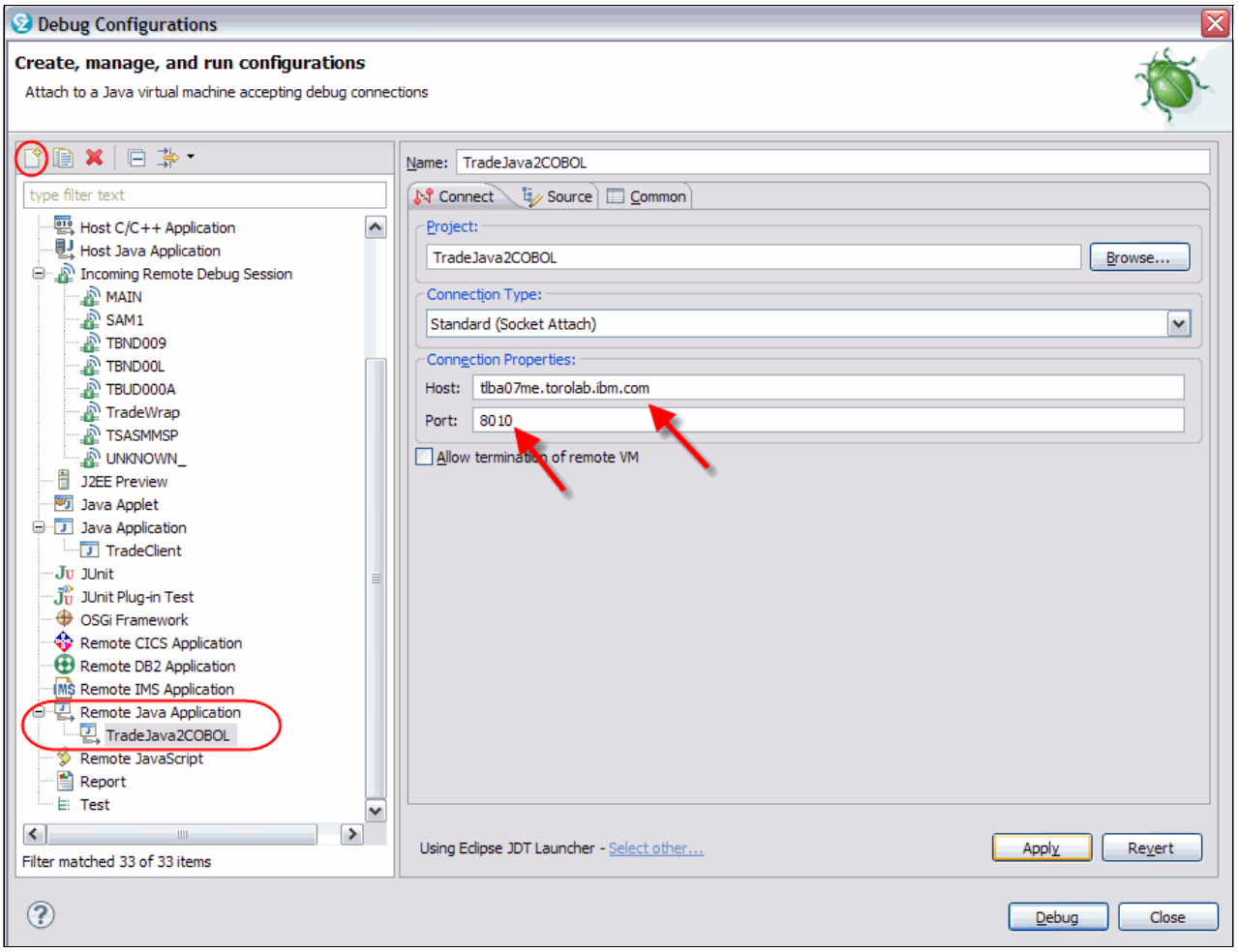


Figure 9-12 Debug Configurations window

3. Select the **Source** tab, as seen in Figure 9-13.

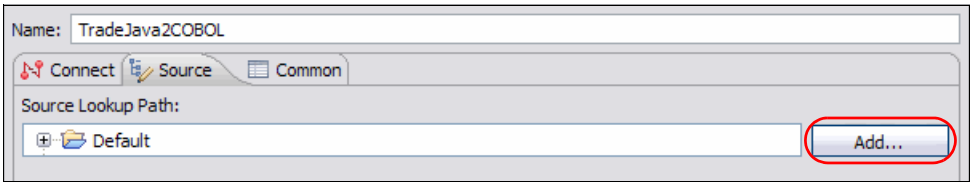


Figure 9-13 Source tab

4. Select **Add**, then select **Java Project**. Click **OK**, as shown in Figure 9-14.

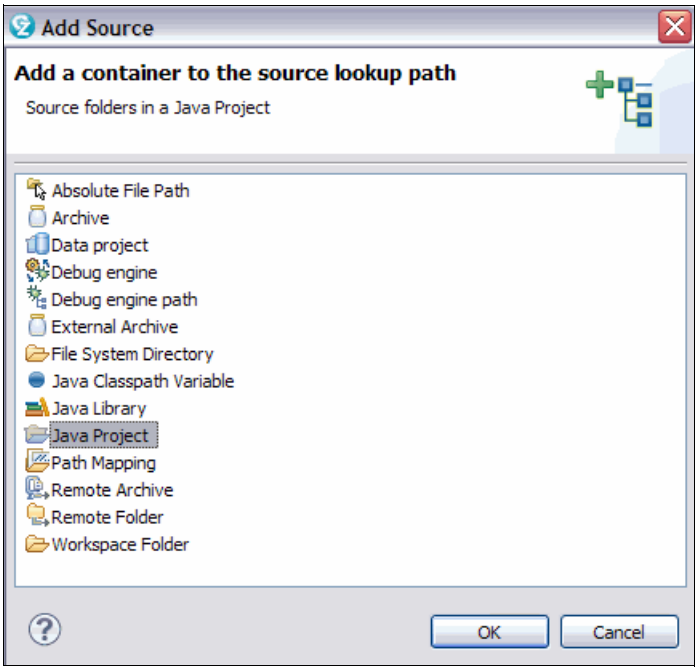


Figure 9-14 Select Java Project window

5. Select **TradeJava2COBOL** and click **OK**, as seen in Figure 9-15.

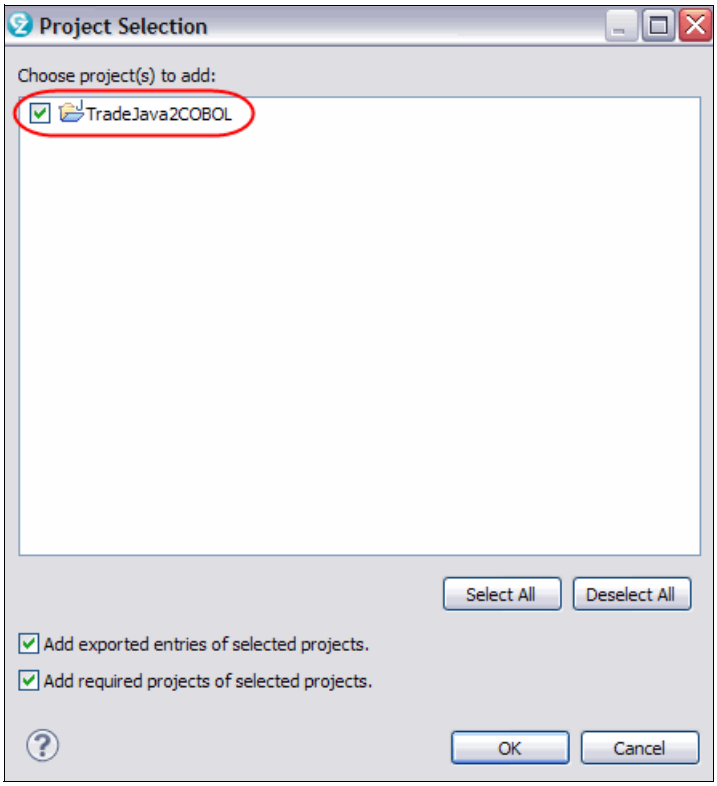


Figure 9-15 Project Selection window

6. As shown in Figure 9-12 on page 293, click **Apply** and **Close**. Do not click **Debug**.

9.2.2 Server (z/OS) set up

After you build the COBOL program on the host, ensure that the JZOS JCL that is started by the system programmer matches the port number. As shown in the following code example, the port number 8010 matches what was specified in Figure 9-12 on page 293. A detailed view of the JZOS environment variable settings is shown Figure 9-16:

```
# Configure JVM options
IJO="-Xms64m -Xmx128m"
IJO="$IJO -Xdebug"
IJO="$IJO -Xrunjdwp:transport=dt_socket,server=y,address=8010"
export IBM_JAVA_OPTIONS="$IJO "
```

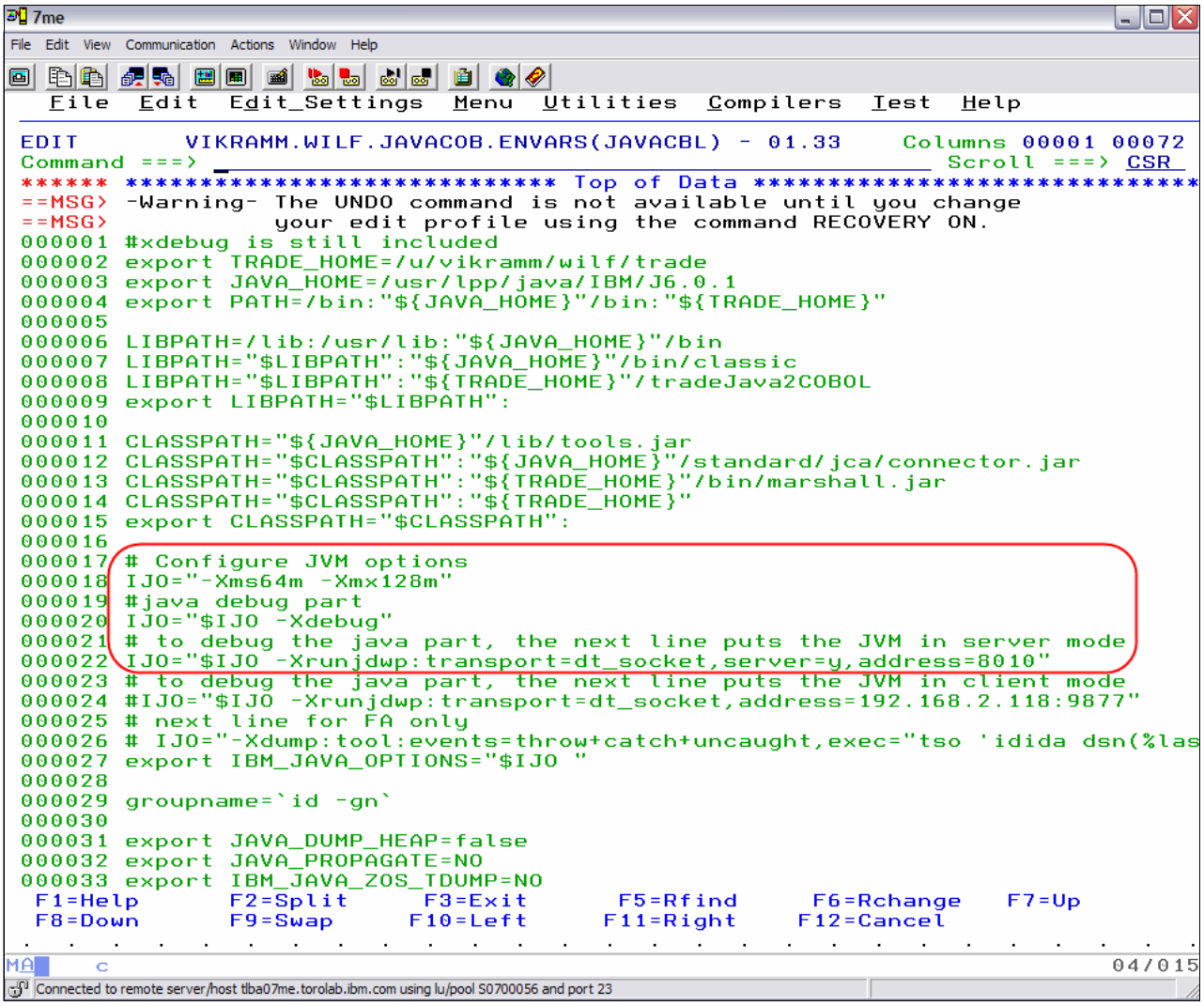


Figure 9-16 JZOS JCL uses these environment variable settings

Debug Tool's Delay Debug Profile on z/OS contains the wanted runtime options (as shown in Figure 9-20 on page 298). The JZOS JCL uses Debug Tool's Delay Debugging support and customized EQAOPTS settings.

Complete the following steps to set up z/OS:

1. Enter B on the Debug Tool Utilities panel. Press Enter (as shown in Figure 9-17).

```
--------------- Debug Tool - Manage Delay Debug Profile Data Set ------
Command ===> _

Specify the name of a delay debug profile data set that you want to
create or edit.

Press Enter to edit the data set.
Press Exit or Cancel to exit.

The data set provides a TEST run-time option for debugging application
and a list program names for pattern matching.


Data Set Name:
  Data Set Name . . . 'VIKRAMM.DLAYDBG.EQAUOPTS'
  Volume Serial . . . _______     (If not cataloged)
```

Figure 9-17 Debug Tool Utilities panel

2. Verify the Delay Debug profile data set. Press Enter (as shown in Figure 9-18).

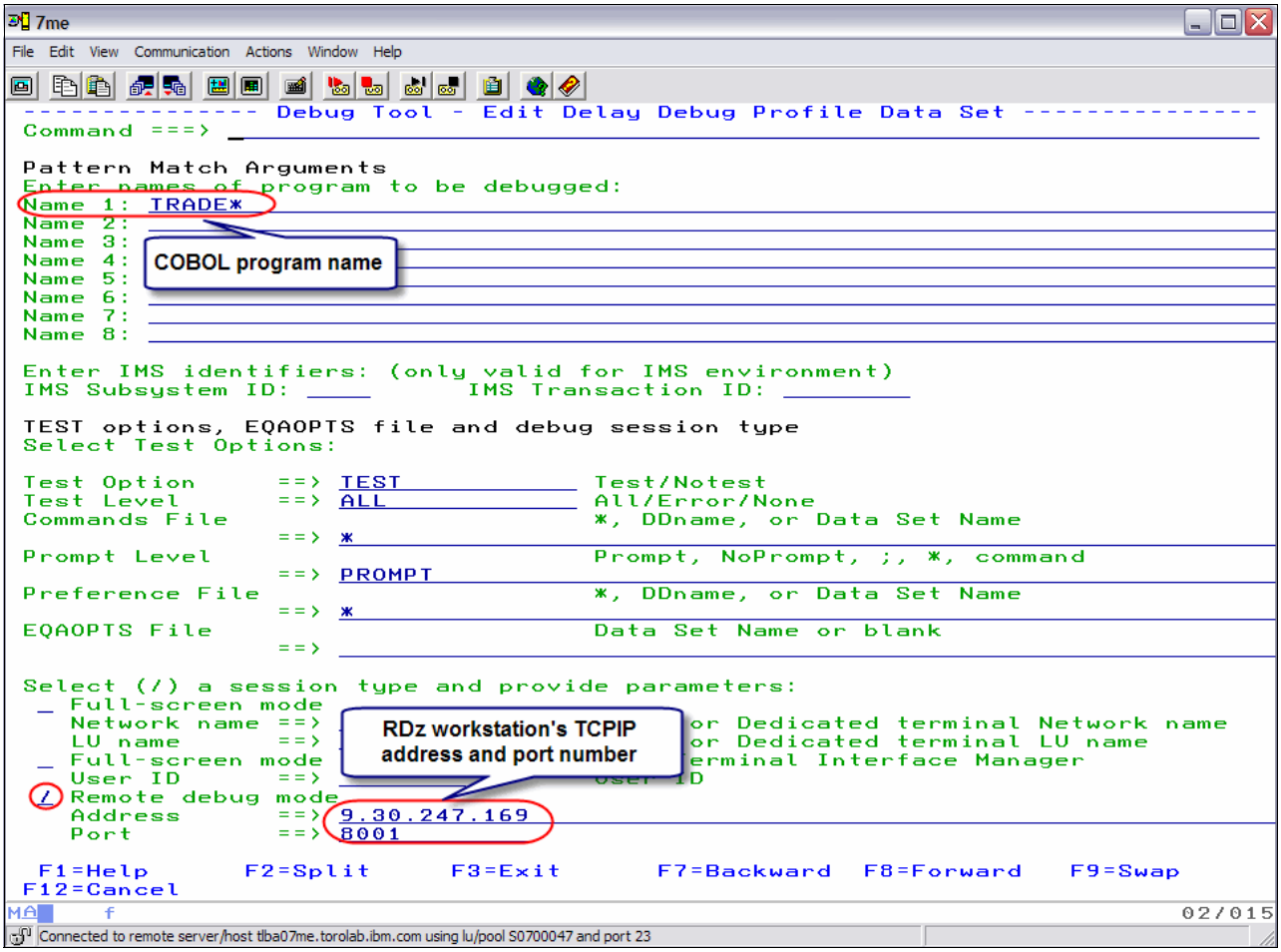


Figure 9-18 Delay Debug profile data set

3. Verify the program name and other TEST runtime options. Press F3 (as shown in Figure 9-19).

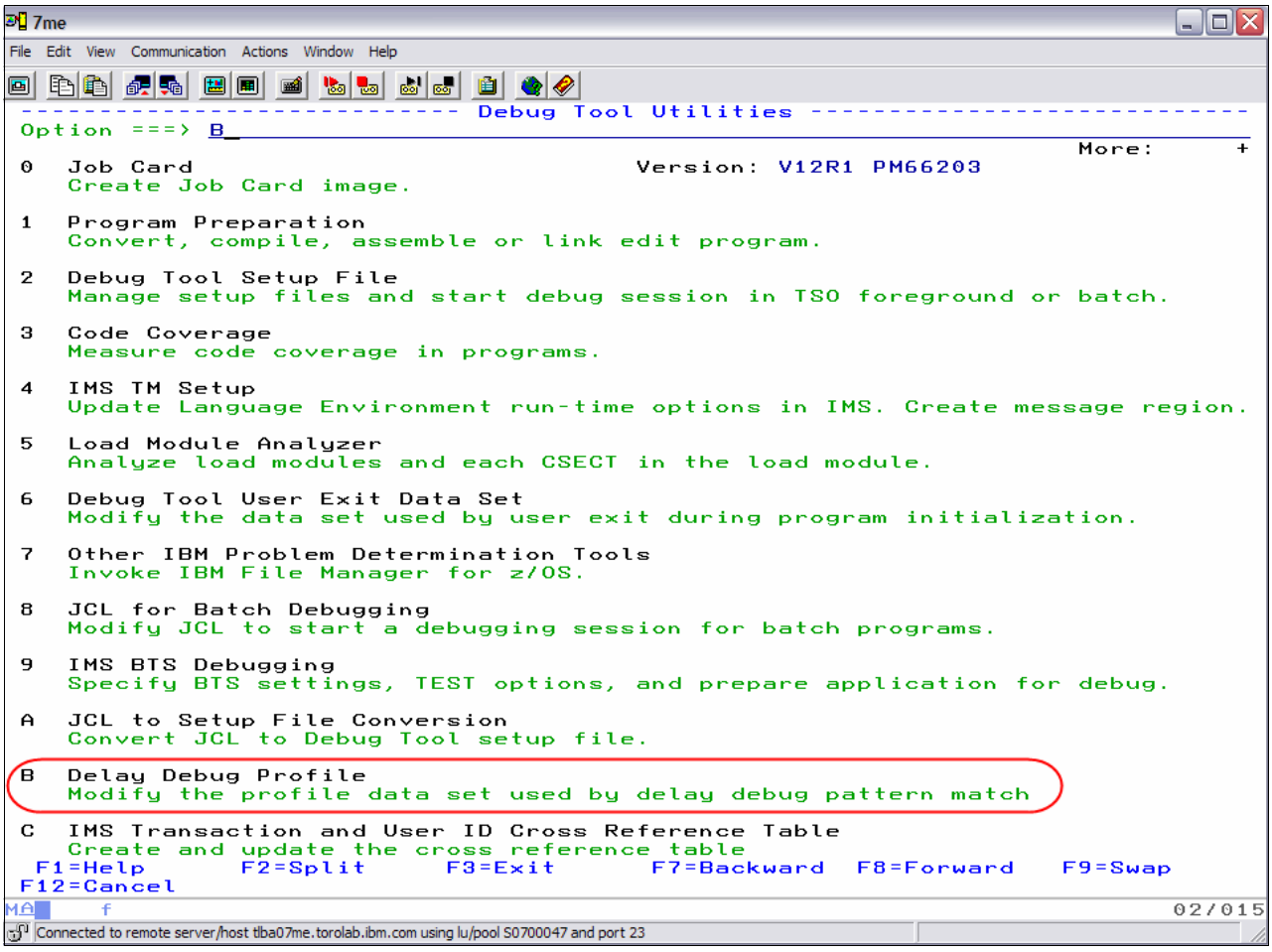


Figure 9-19 Program name and other TEST runtime options

9.2.3 COBOL programs on z/OS

As seen in Figure 9-20, the COBOL program TradeWrap links in a COBOL program, DLL2NDLL, which starts another COBOL program, TRADE2.

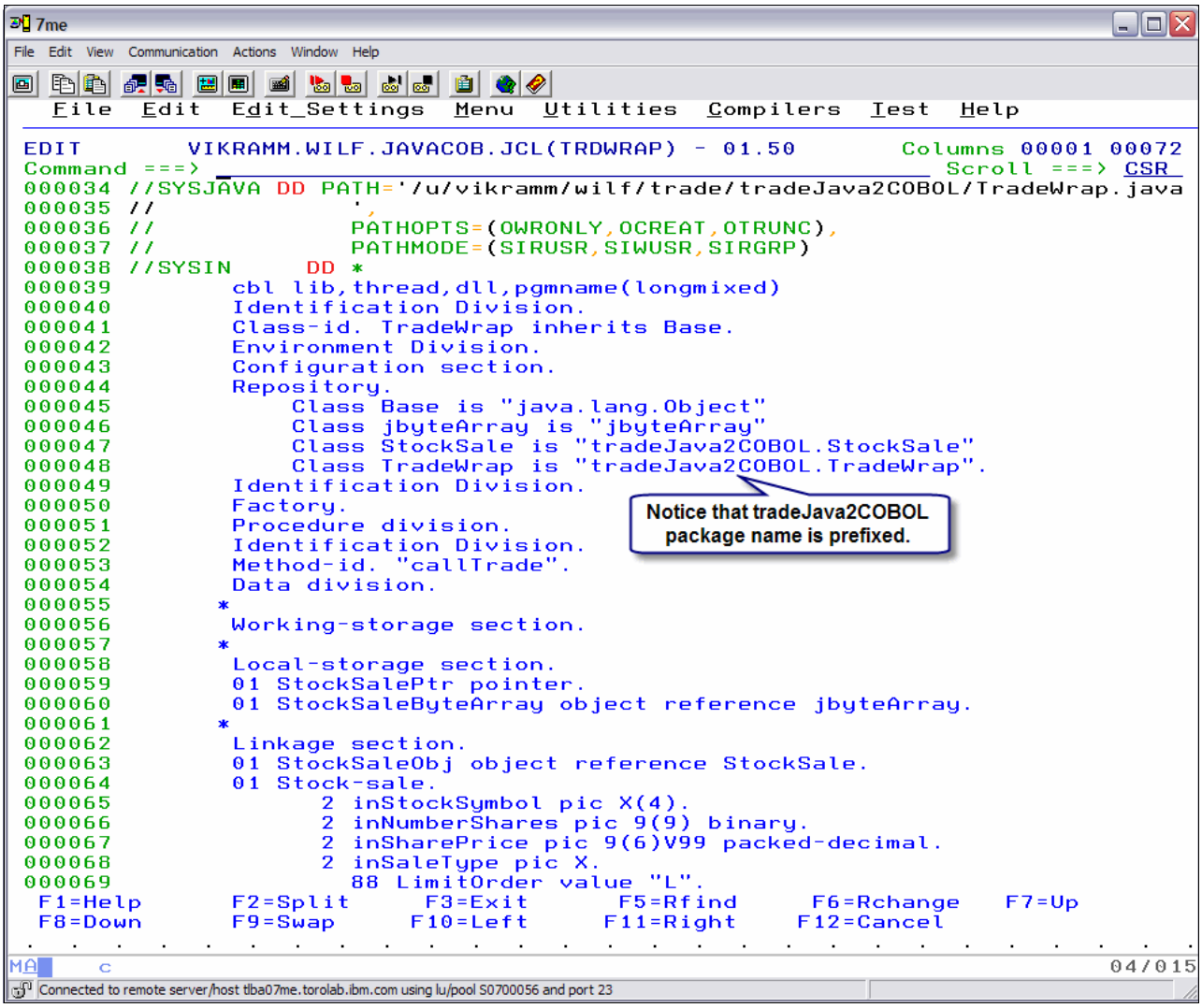


Figure 9-20 TradeWrap COBOL program generates the TradeWrap.java file

The TradeWrap COBOL program generates the `TradeWrap.java` program, which can be imported (in binary form) into the Java project that we saw in Figure 9-10 on page 291.

9.2.4 End-to-end debugging starts and ends in Java

Complete the following steps to start the Debug tool:

1. Set two line breakpoints in `TradeClient.java` (as shown in Figure 9-21 on page 299).
2. Click **Debug** in the TradeJava2COBOL Debug Configuration (as shown in Figure 9-12 on page 293).
3. Debug the Java code (as shown in Figure 9-22 on page 300).
4. Debug the COBOL code (as shown in Figure 9-23 on page 301 and Figure 9-24 on page 302).
5. Debug the Java code (as shown in Figure 9-25 on page 303).

Complete the following steps to start the Debug tool:

1. As shown in Figure 9-21, set a line breakpoint in `TradeClient.java` program by double-clicking in the source prefix area.

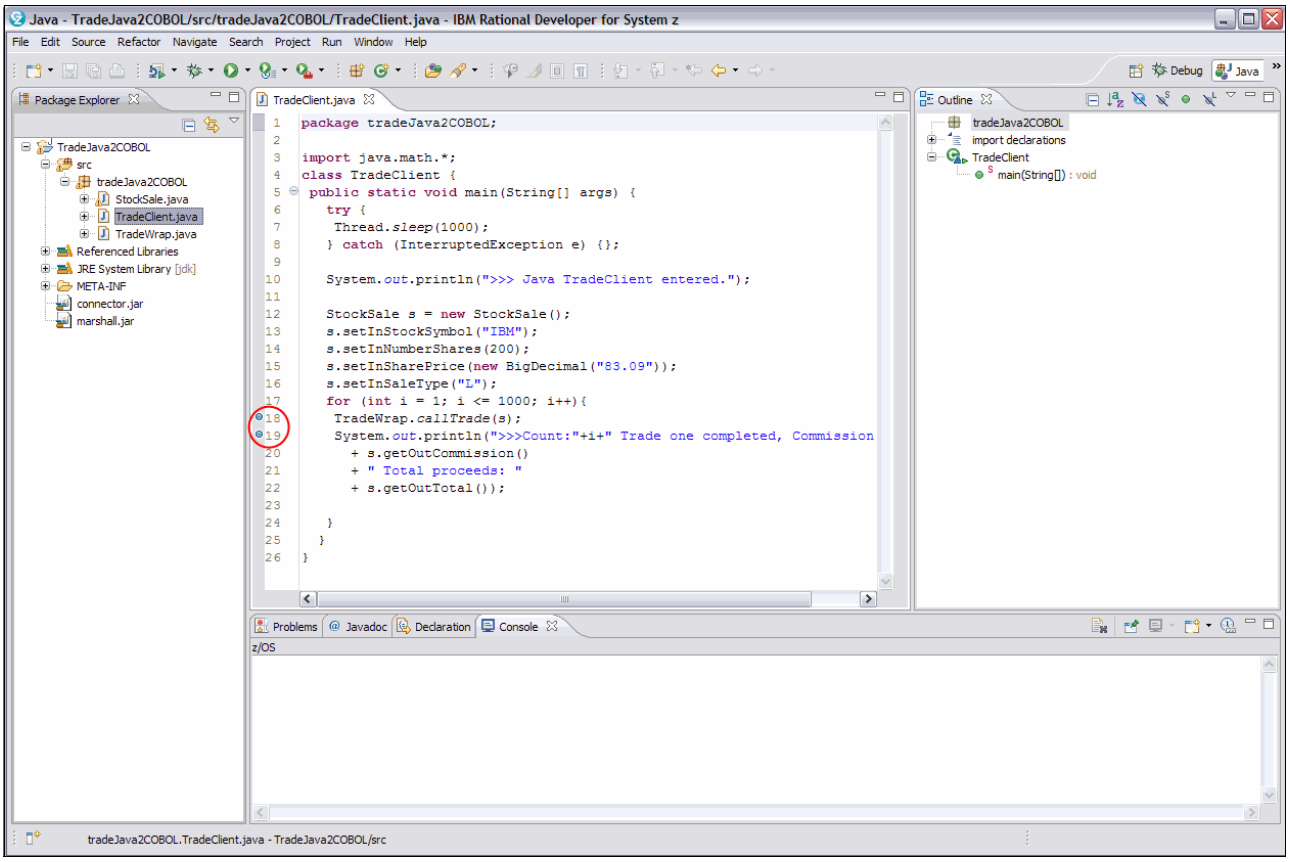


Figure 9-21 Set two line breakpoints in TradeClient.java window

2. Open the TradeJava2COBOL debug configuration (as shown in Figure 9-12 on page 293) and click **Debug**.
3. Stop at the line breakpoint in the `TradeClient.java` program that is shown in Figure 9-22 on page 300. Click **Resume** (or press F8).

> **Resume icon:** The Resume icon is the green arrow in the Debug view toolbar. It is circled in green in Figure 9-22.

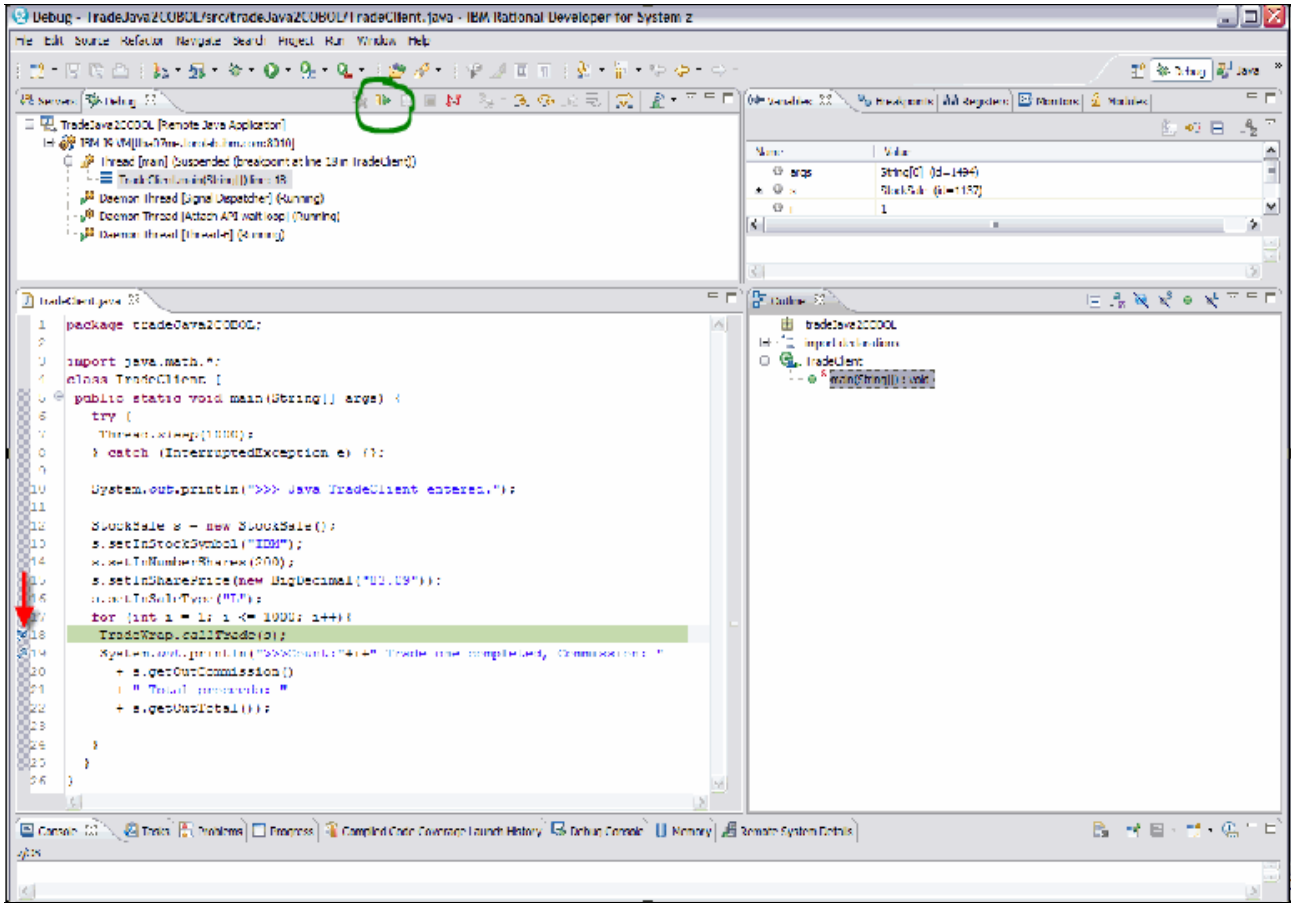


Figure 9-22 Stop at the line breakpoint in the TradeClient Java program window

4. Debug the COBOL code by using Debug Tool that is shown in Figure 9-23. In section 4.4.4, “Debugging COBOL, PL/I, Assembler, and C++ programs” on page 133, the debugging actions that the reader could perform for debugging the COBOL code are described.

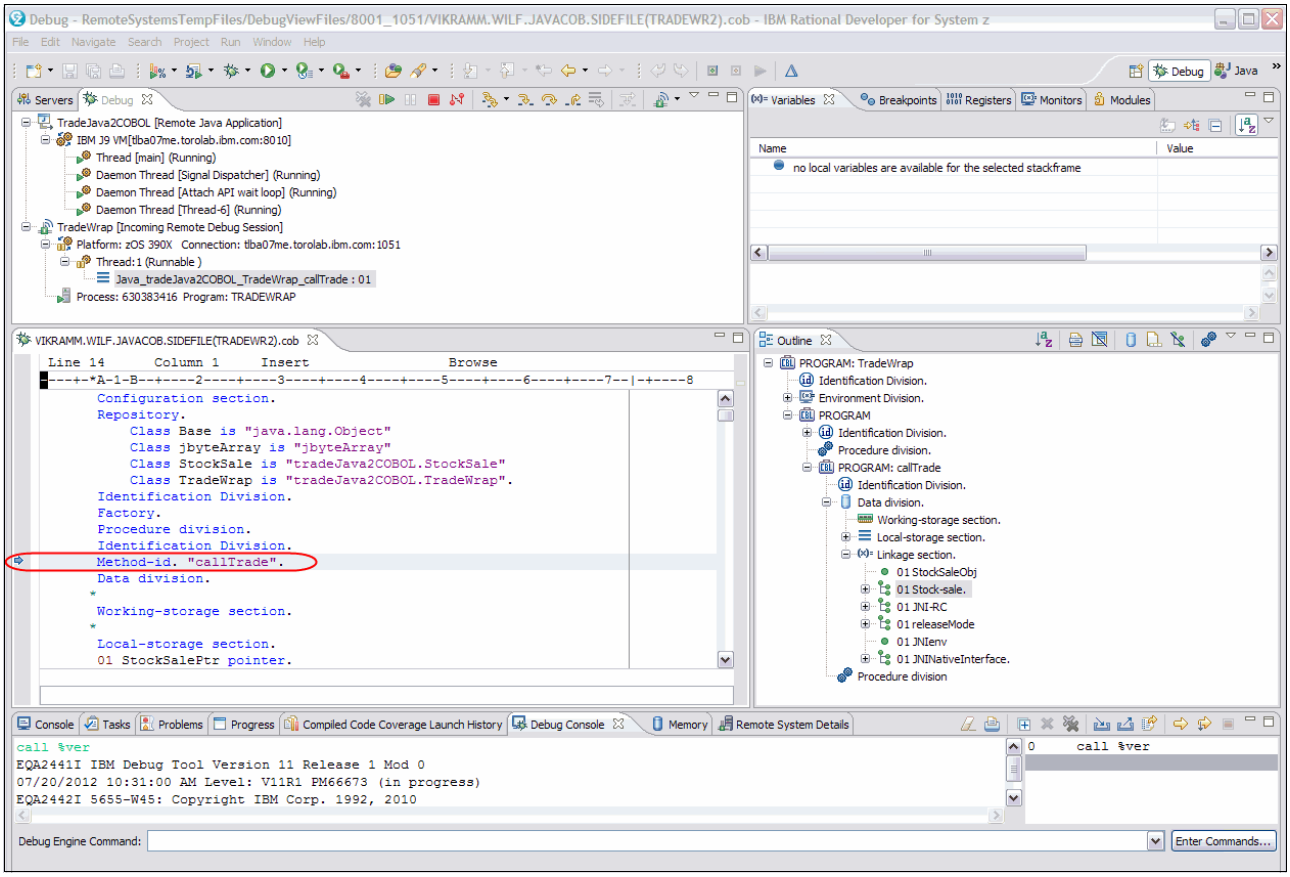


Figure 9-23 Debug COBOL code

5. Click **Resume** after debugging TRADE2 COBOL program.

> **Resume icon:** The Resume icon is the green arrow in the Debug view toolbar. It is circled in green in Figure 9-24.

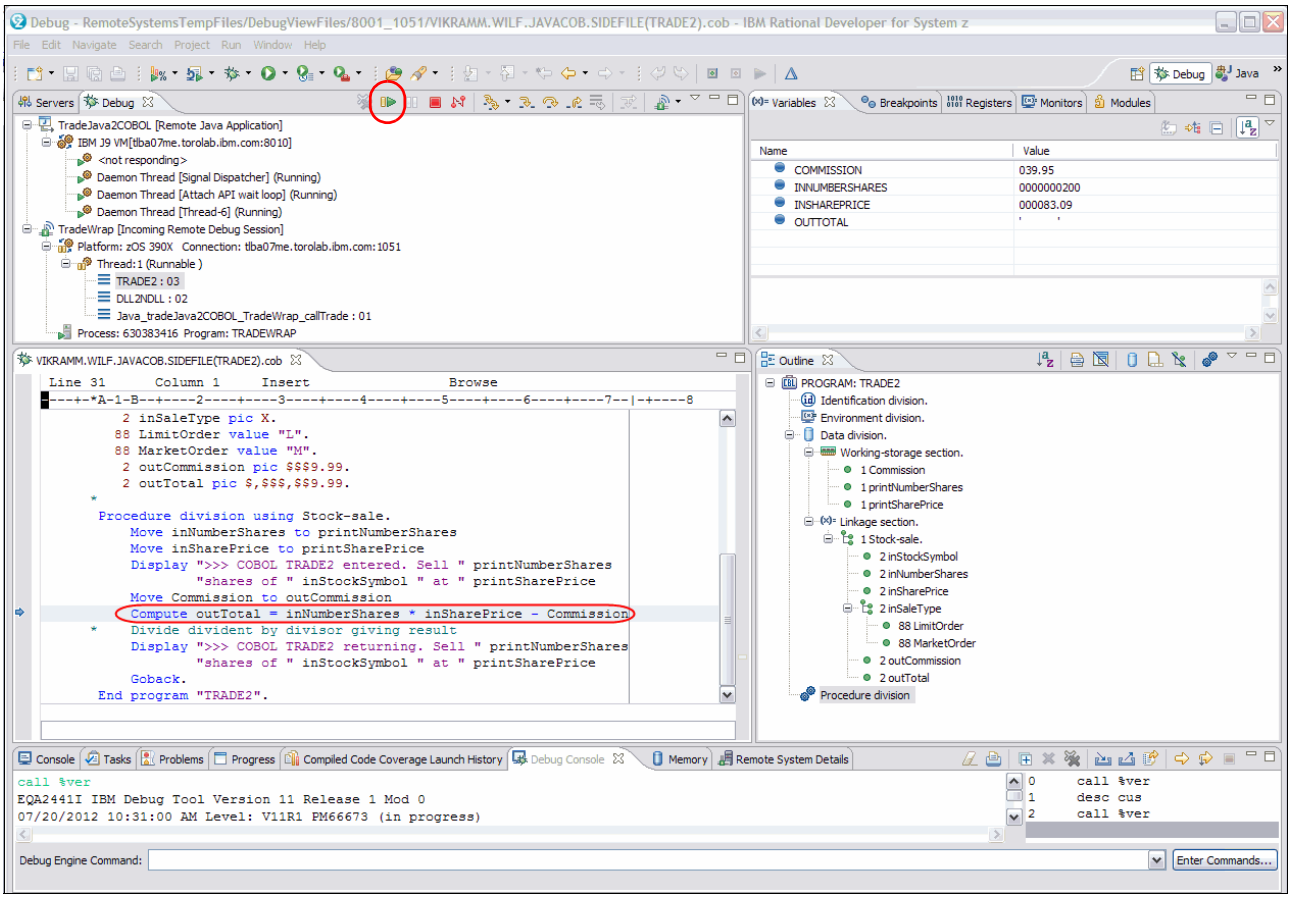


Figure 9-24 Business logic from TRADE2 COBOL program

The Debug tool is stopped in the Java application because of the line breakpoint on line 19. You can click the **Disconnect** icon (which is circled in red in Figure 9-25) in the Debug view toolbar to end the debug session.

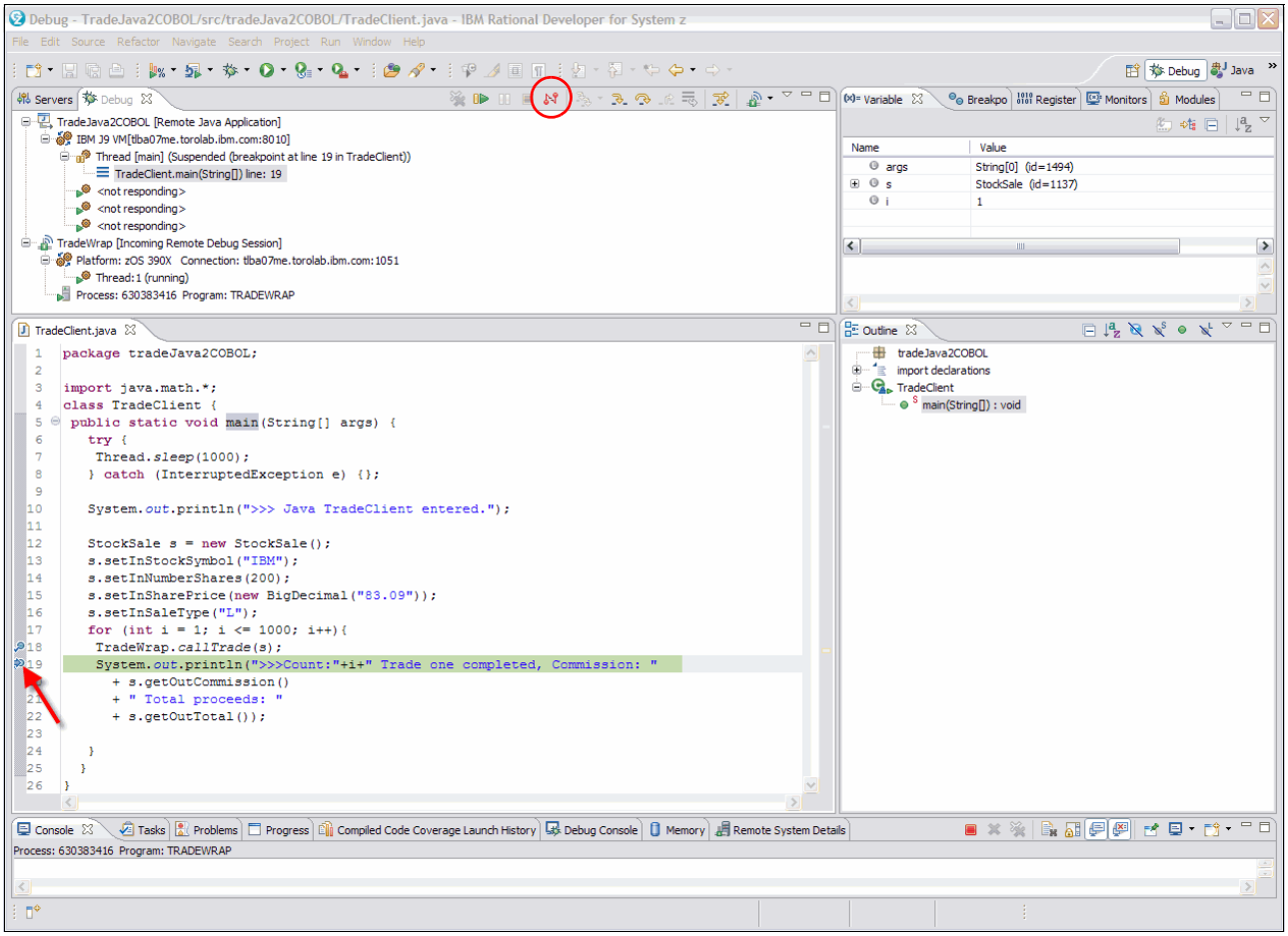


Figure 9-25 Stop back in the Java application window: Line breakpoint at line 19

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics that are covered in this book.

IBM Redbooks publications

The following IBM Redbooks publications provide additional information about the topics in this book. Note that some publications referenced in this list might be available in softcopy only:

- *IBM Application Development and Problem Determination Tools V7 for System z: Application Performance Analyzer, Debug Tool Utilities and Advanced Functions, Fault Analyzer, File Export, File Manager, and Workload Simulator*, SG24-7372
- *IBM Application Development and Problem Determination*, SG24-7661
- *CICS Explorer*, SG24-7778
- *Extend the CICS Explorer: A Better Way to Manage Your CICS*, SG24-7819

You can search for, view, download, or order these documents and other Redbooks publications, Redpaper publications, Web Docs, draft, and additional materials at this website:

`http://www.ibm.com/redbooks`

Online resources

The following websites are also relevant as further information sources:

- Problem Determination Tools plug-ins software download sites:
  - `http://www-01.ibm.com/software/htp/cics/explorer/download/`
  - `http://www-01.ibm.com/software/awdtools/deployment/pdtplugins/`
- Problem Determination Tools libraries:
  - `http://www-01.ibm.com/software/awdtools/apa/library/`
  - `http://www-01.ibm.com/software/awdtools/debugtool/library/`
  - `http://www-01.ibm.com/software/awdtools/faultanalyzer/library/`
  - `http://www-01.ibm.com/software/awdtools/filemanager/library/`
  - `http://www-01.ibm.com/software/awdtools/workloadsimulator/library/`

IBM Application Performance Analyzer for z/OS

The following documents are included with the product:

- *User's Guide Version 12 Release 1*
- *Customization Guide Version 12 Release 1*
- *Messages Guide Version 12 Release 1*

Help from IBM

IBM Support and downloads:

http://www.ibm.com/support

IBM Global Services:

http://www.ibm.com/services

IBM Problem Determination Tools for z/OS

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

IBM Problem Determination Tools for z/OS

Fault Analyzer demonstrations

Debugging demonstrations

Application Performance Analyzer demonstrations

IBM Problem Determination Tools consists of a core group of IBM products that are designed to work with compilers and run times to provide a start-to-finish development solution for the IT professional. This IBM Redbooks publication provides you with an introduction to the tools, guidance for program preparation to use with them, an overview of their integration, and several scenarios for their use.

If an abend occurs during testing, Fault Analyzer enables the programmer to quickly and easily pinpoint the abending location and optionally, the failing line of code. Many times, this information is all the programmer requires to correct the problem. However, it might be necessary to delve a little deeper into the code to figure out the problem. Debug Tool allows the programmer to step through the code at whatever level is required to determine where the error was introduced or encountered.

After the code or data is corrected, the same process is followed again until no errors are encountered. However, volume testing or testing with multiple terminals is sometimes required to ensure real-world reliability. Workload Simulator can be used to perform this type of testing.

After all of the tests are completed, running the application by using Application Performance Analyzer can ensure that no performance bottlenecks are encountered. It also provides a baseline to ensure that future enhancements do not introduce new performance degradation into the application.

This publication is intended for z/OS application developers and system programmers.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7918-02 0738437506