Redbooks
ibm.com/redbooks

IBM CICS Interdependency Analyzer

Em James

z Systems

IBM

Redbooks

International Technical Support Organization

IBM CICS Interdependency Analyzer

December 2015

SG24-6458-02

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

Third Edition (December 2015)

This edition applies to Version 5, Release 3, CICS Interdependency Analyzer.

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 grant 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:

CICS®	IMS™	Redbooks (logo) ®
CICS Explorer®	Language Environment®	RMF™
CICSPlex®	MVS™	System z9®
ClearCase®	OS/390®	WebSphere®
DB2®	RAA®	z Systems™
IA®	Rational®	z/OS®
IBM®	Redbooks®	z9®

The following terms are trademarks of other companies:

Microsoft, Windows, Windows NT, 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.

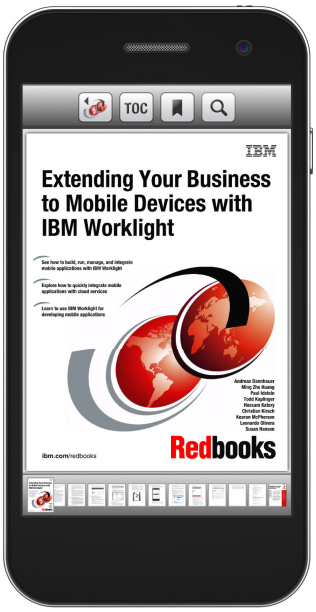
Other company, product, or service names may be trademarks or service marks of others.

Find and read thousands of IBM Redbooks publications

- Search, bookmark, save and organize favorites
- Get up-to-the-minute Redbooks news and announcements
- Link to the latest Redbooks blogs and videos

Get the latest version of the Redbooks Mobile App

iOS **Download Now** Android



Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!

ibm.com/Redbooks
About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

Preface

The IBM® CICS® Interdependency Analyzer (CICS IA®) is a runtime tool for use with IBM CICS Transaction Server for z/OS®. CICS IA allows both system programmers and application developers to get an understanding of the relationships and dependencies of your CICS applications and the environment on which they run. By analyzing data collected by CICS IA, you can make changes to your environment in a safe and controlled but timely manner to address changing demands on your business applications.

In this IBM Redbooks® publication, we first provide a detailed overview of what CICS IA is and what business issues it addresses before we review how to configure CICS IA to collect the data that you require with the minimum provenance impact. We then show how you can analyze this data to assist with day-to-day application changes and major projects such as application onboarding.

Author

This book was produced by Em James at the International Technical Support Organization, Hursley Center.

Em James is a Technical Specialist for CICS Tools working at IBM Hursley Labs in the United Kingdom. He has 30 years of experience working with CICS as both an Application Programmer and a Systems Programmer. He most recently worked as the Lead Developer on the CICS IA product. He has a degree in Computer Science from Loughborough University.

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

Authors of the first edition, IBM CICS Interdependency Analyzer, published in November 2004, were:

- Thomas Braun
- James Matson
- Dave White

Authors of the second edition, IBM CICS Interdependency Analyzer, published in September 2008, were:

- Chris Rayns
- Peter Klein
- Juerg Rudolf
- Andy Wright

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:

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 Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  https://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

In this section, we describe the technical changes that we 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 SG24-6458-02 and IBM CICS Interdependency Analyzer,
as created or updated on December 8, 2015.

December 2015, Third Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

Deleted information

- Chapter 2 Installation and Customization
- Chapter 3 Scanner Component
- Chapter 4 The collector
- Chapter 5 Query Interface
- Chapter 6 Analyzing IA data using the CICS IA Explorer
- Chapter 7 Sample Reports and Queries
- Chapter 8 CICS IA usage scenarios
- Chapter 9 Affinities
- Chapter 10 Hints and Tips
- Chapter 11 Debugging

New information

- Chapter 2 Preferred practices
- Chapter 3 Using CICS IA daily
- Chapter 4 Using CICS IA data for affinity analysis
- Chapter 5 Using CICS IA data for threadsafe analysis
- Chapter 6 Application onboarding
- Chapter 7 Modernization with CICS Events
- Chapter 8 Command Flow feature
- Appendix B Task collection frequency: Performance results

Changed information

- Chapter 1 Introduction
- Appendix A IBM Rational® Asset Analyzer

September 2008, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

New information

- Chapter 6 CICS IA Explorer
- Chapter 8 CICS IA usage scenarios
- Chapter 10 Hints and Tips

Changed information

- Chapter 1
- Chapter 2
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 7
- Chapter 9
- Chapter 11

1

IBM CICS Interdependency Analyzer overview

In this chapter, we provide an overview of CICS Interdependency Analyzer (IA). This chapter contains the following sections:

In this book, we provide information about a number of different CICS IA usage scenarios. We cover both how CICS IA can assist daily and on major projects including:

- Day to day use: Impact analysis of code changes
- Understanding CICS region affinities and creating IBM CICSPlex® System Manager (CPSM) transaction groups for workload management policies
- CICS threadsafe analysis
- Application onboarding
- Application modernization using CICS events

1.1 What CICS IA can do for you

There are many business reasons for using CICS IA, and they vary by industry. This section contains some of the business imperatives that face corporations today.

Day-to-day maintenance or enhancement of applications

During the normal application lifecycle, CICS applications require maintenance and enhancement. When a programmer who is unfamiliar with the application that they are required to modify starts the modification process, much time can be spent in trying to understand the application and the inherent flow of transactions. This learning curve can be greatly reduced through the use of CICS IA, which identifies the resources that are affected directly and indirectly.

Upgrading your CICS TS version to the latest release

Upgrading your CICS TS systems to the latest release is a significant piece of work for most customers. The testing of applications against the latest release is a major part of this work. CICS IA can identify transactions and programs that use CICS APIs that have been modified or removed in the latest release. CICS IA also provides information about the CICS global user exits (GLUEs) and task-related user exits (TRUEs) used by each CICS region allowing you to work with product vendors to ensure support at the latest release.

Optimizing your CICS regions

In the latest releases of CICS TS, much work has been done to enable you to optimize your CICS regions. In V5 of CICS, many more of the commands are now threadsafe allowing more of your application programs to run on the open task control blocks (TCBs). CICS IA provides you with a Threadsafe report that can assist you in identifying which programs can take advantage of this feature. The CICS command flow feature also assists with threadsafe analysis by showing you the commands that introduce TCB swaps. Also, in CICS TS V5, work has been done to reduce the amount of 24-bit storage used by CICS including the ability to move the Global Work Area associated with your CICS exits to the 31-bit addressing. CICS IA can help you understand where your programs get storage and also what exits you use.

Modernizing your applications

Recent releases of CICS introduced many new resources and commands including Platform and Applications, CICS Events, and atomservices. CICS IA collects information for all of these new features. CICS IA tells you what resources are used by your “Application” and in which “Platform.” It identifies where CICS Events are triggered and assists in identifying where you want to create an event. All new commands and resources are captured by CICS IA from day one of a new CICS TS release.

CPSM deployment

One of the major tasks when you have deployed CPSM, is to take advantage of the *workload management* (WLM) feature. In order to adopt WLM and dynamically route your work across your target regions, you first need to understand your CICS “Affinities” that your transactions might have. CICS IA provides a specialist collector for identifying your Affinities, reporting them by affinity type, and the ability to built CPSM WLM Transaction groups that can be assigned to your WLM policy.

Mergers and acquisitions

Many banks are involved in mergers and acquisitions. The result is that they have to consolidate workloads and move CICS applications around for isolation reasons or to spread the workload for performance reasons. Because these applications might not be well understood by the acquiring bank and documentation might be inadequate, there is a need to understand all of the resources that are associated with a given application.

Outsourcing

Large outsourcing companies are continually facing the problem of running CICS applications with which they are unfamiliar. Often, naming standards are lacking or conflict with other applications that are running on the same logical partition (LPAR). Documentation might be nonexistent or incomplete. Again, there is a need for a tool to facilitate the understanding of the resource interdependencies and affinities that are involved.

1.2 CICS IA highlights

Here are some of the benefits of using CICS IA:

- Automate collection of runtime resource relationships
- Impact analysis of routine application maintenance for the developer
- Understand application flow with flexible resource-relationship reports

- Compare applications and resources across regions and platforms
- Upgrade your CICS TS release quicker
- Deploy WLM policies with confidence that you have no affinities
- Build your own SQL queries to answer your questions on your CICS resource relationships
- Support service-oriented architecture (SOA) implementations with deep application understanding
- Help you identify and deploy CICS Application Events
- Visualize your resource usage by region, transaction, or program
- Track your work load from start to finish including “CICS transaction data”

1.2.1 Questions that CICS IA answers

In this section, we provide some of the questions that CICS IA can answer, and we offer them to give you a sense of what is possible with CICS IA:

- What region does a particular CICS program run in?
- What are all of the CICS resources that a given transaction can use?
- What programs do a given transaction invoke?
- What transactions access a particular file and how?
- What resources do a specific program use?
- How is a file accessed by a particular program?
- Which affinities do a transaction have?
- How do you query on IBM DB2®, IBM MQ, or IBM IMS™ resource use?
- What resources are used by a Webservice?
- In which CICS region is a Webservice deployed?
- Does this program issue a COBOL display command?

And more recently with the introduction of CICS TS V5 Platforms and Applications, these questions apply:

- What platform is your application deployed in?
- What are all of the CICS resources that a given application uses?
- In which regions is an application deployed?
- What resources are used by a specific application version?
- What are the resource relationships for different application versions?

1.3 CICS IA

IBM CICS Interdependency Analyzer for z/OS is the IBM discovery tool for CICS Transaction Server for z/OS, which completely supports the new functionality in the latest versions of CICS Transaction Server.

CICS IA provides software for both a runtime collector and a batch job for use with your CICS Transaction Server for z/OS environments. Together, these features provide you with a discovery tool that allows you to fully understand the CICS resource relationships within your environment.

The runtime collector comes in three different varieties.

The Dependency Collector

This function collects information about the majority of the CICS commands executed and the associated resources. The information is collected by CICS region, transaction ID, and program name. In CICS TS V5 and above it can also collect data by Platform and Application.

With time, this function has been enhanced to capture:

- DB2 commands and resources
- IBM MQ commands and resources
- IMS commands and resources (old and new)
- CPSM commands
- Software AG Natural calls and ADABAS calls within a Natural program
- Detailed information for:
  - Programs
  - Transactions
  - Webservices
  - Exits
  - Files
  - TDQueues
  - TSQueues
  - Connections (MRO / IPIC)

CICS IA captures interdependency information while CICS is running via a number of CICS exits (GLUEs and TRUES) and stores this information in a data space. This data eventually ends up in a DB2 database (via VSAM files).

The data stored in this database enables you to understand the makeup of your application set, such as:

- Which transactions use which programs
- Which programs use which resources (files, maps, queues)
- What resources are used by a Webservice

- Which applications are deployed in this platform
- Which programs use the IBM MQ queue
- What CICS task-related user exits do I use in a region

The Affinity Collector

This feature is a more specialized collector in that it captures information about CICS Affinities. It captures both Transaction System and Inter Transaction affinity types.

Note: Why do you need to identify Affinities?

Affinity information is useful in a dynamic routing environment because you need to know of any restrictions that *prevent* particular transactions from being routed to particular application-owning regions (AORs) or that *require* particular transactions to be routed to particular AORs. This analysis is especially true when first setting up a WLM policy in CPSM.

Again, CICS IA loads the Affinity data into its DB2 databases. You can then query the data with the CICS IA Explorer, with batch SQL queries, or with other SQL query software tools.

For more information about CICS IA and affinities, see Chapter 4, “Using CICS IA data for affinity analysis” on page 81.

The Command Flow Collector

This feature was introduced in CICS IA V3.1. Again, it collects all of the EXEC CICS commands, DB2, IBM MQ, and IMS calls but in a chronological order. You can think of it as a command trace.

It was initially designed for use by a single systems programmer to assist with threadsafe analysis. As well as capturing the commands, we captured the TCB mode at which the command was executed. This information enables you to quickly establish the flow of your transaction across the available TCB modes and which commands cause it to use the QR TCB.

It was later expanded to be a multi-user feature and is now used by many application developers to understand the flow of the application for a given set of parameters. It has also been enhanced to capture the CPU time before and after each command, the actual TCB switch count, and transaction tracking data such as point of origin.

Unlike the dependency and affinity collector, the data is written straight out to a user journal, which is then offloaded to a generation data group (GDG) and then loaded into the DB2 database. The use of user journals allows the feature to be configured so that each developer has its own journal, or more common, their own GDG.

For more information about the Command Flow Collector, see Chapter 8, “Command Flow feature” on page 197.

1.3.1 The history of CICS IA

CICS IA started life as a support pack. Initially, it was designed to run in a single region and to capture the traditional CICS resources: Transactions, programs maps, files, TDQueues, and TSQueues. The collected data was stored in VSAM files and a batch report of your resources used was available. A DB2 back-end was added to this support pack and the first release was shipped in 2001.

There have been many releases and enhancements since then. Currently, a new CICS IA version is available at the same time as a new CICS TS release. From V5 onwards, the CICS IA and CICS TS version numbers match. The latest release always tolerates the latest CICS release and also captures any new commands and resource data added. There follows a historical list of the CICS IA releases and the enhancements made. It shows that CICS IA is very much an “active” product and is continuously enhanced to meet customers requirements.

Enhancements in CICS IA Version 5 Release 3

The following enhancements were made in CICS IA Version 5 Release 3.

Scenario-based collections

Previously in CICS IA, it was possible to operate the Dependency and Affinity collector from the CICS IA plug-in. This capability is now extended to include scenario-based collections where the CICS IA options that are required are set, depending on the type of collection required. For example, you can start a CICS Threadsafe Collection. Selecting to run a specific scenario turns on all of the options that are required for that scenario before the collector is started.

New CICS IA perspective

Within the CICS IA perspective (plug-in), there are fewer views. There is a new IA Navigation view under which you can find different folders that contain all of the data. This new consolidated view eliminates the need to switch between the views in the middle pane. Some other views are transformed to editors that are only displayed on-demand to improve the usability and overall user experience within the perspective.

Default configuration

The CICS IA configuration utility is updated to allow the user to select a “DEFAULT” or a “FULL” configuration. The “DEFAULT” configuration prompts you to enter the minimum values that are required to get your CICS IA system up and running. To fine-tune the default values within your environment, select the “FULL” configuration.

Database purge

The CICS IA database purge tools are extended. You can now delete:

- TSQ or ENQ objects, which are related resources with a duplicate name prefix
- Records that are related to old versions of a program

Other enhancements

Other CICS IA Version 5.3 enhancements include the following:

- The collection of dynamic calls for COBOL Version 5.
- The identification of the remote SYSID and remote name for file commands.
- The Resource Prefix list is replaced by the Resource Compression list, which allows the user to apply a compression rule and also preserve a part of the resource name.
- The command flow feature is expanded to collect Software AG’s Natural program calls.

Enhancements in CICS IA Version 5 Release 2 (2014)

The following enhancements were introduced in this release:

- The dependency collector now lets you reduce the performance impact of running the exits by collecting on a reduced number of tasks. The “Trigger for Task Collection” option has been added to the General options panel to provide this facility. For more information, see “Collecting CICS IA data in a production region” on page 24.
- Information on the resources collected is now written to the CIULOG.
- A new Collector Statistics menu is provided to assist you to determine how much data and what type of data you have collected.
- The Affinity report and build can now be run from the CICS IA plug-in.
- A batch job can be shipped from the plug-in to create CPSM Transaction Groups from the builder output.

- The threadsafe analysis report has been improved by collecting the following:
  - CPSM API calls
  - Connection details (IPIC or MRO)
  - Identifying reentrant load modules
- The support for the CICS TS Application technology has been enhanced to include CICS TS platform support:
  - The CICS IA Version 5.2 exits capture platform information and associates it with Dependency and Command Flow data that is collected.
  - The CICS IA Version 5.2 plug-in has new and updated views to support this feature.
- The CICS IA Version 5.2 plug-in provides the following new “timeline”-based views that show you:
  - TCB mode switches
  - Switches across regions
  - Application and platform switches
- The CICS IA plug-in provides a new view so that you can compare program attributes across regions.

Enhancements in CICS IA Version 5 Release 1 (2012)

The following enhancements were introduced in this release:

- Support for the CICS TS Application technology introduced in CICS TS V5.1.
- The CICS IA exits capture Application information and associates them with Dependency Data and Command Flow Data collected.
- The CICS IA configuration execution is split into two tasks allowing you to:
  - Configure the target DB2 Environment.
  - Configure the CICS regions in which you want to collect data.
- CICS IA installation and configuration “cheat sheets” provided with the CICS IA plug-in.
- The CICS IA threadsafe analysis report can be run in the CICS IA plug-in.
- You can now delete your CICS IA data by “collection ID” from the CICS IA plug-in or in batch.
- Support for Native SQL language stored procedures for use with DB2 V9.1 and later. They are eligible to be run in an IBM System z9® Integrated Information Processor (zIIP) if one is available.

CICS IA V5.1 with PTFs UK94793, UK94794, and UK94795 for APAR PM82414 applied delivers the following new capabilities:

- The IA configuration allows you to define your site's SORT utility and your system DUMP High-Level Qualifier
- Timestamps were added to the command flow data
- Command Flow collector supports the transaction and program exclude lists.
- The COBOL display command is captured as part of the dependency collection

Enhancements in CICS IA Version 3 Release 2 (2011)

The following enhancements were introduced in this release:

- The Command Flow feature was enhanced to allow multiple users to capture individual command flow traces.
- A new CINC transaction is used to operate and administer the Command Flow collection
- You can now control the operation of the CICS IA Dependency and Affinity Collector by using the CICS IA plug-in for the IBM CICS Explorer®. You can START/STOP/PAUSE/CONTINUE and REFRESH the controller.
- The dependency collector now identifies the program invoking CICS, IBM MQ, IMS, DB2 commands, even if this program is a dynamically called program.
- Affinity data can now be extracted to CSV files for use with DB2 UDB databases.
- New IBM MQ V7.1 API commands collected
- ISPF configuration values can be shared across users
- DB2 table and views added to allow you to map a CICS exit to a product name and description
- Improved diagnostics is provided via the use of the CICS TS user trace feature. This feature allows up to three levels of CICS IA tracing
- "Collection ID" added to the dependency collection at DB2 load time. This identifier enables the user to load, manage, and compare resource usage by collection ID

Enhancements in CICS IA Version 3 Release 1 (2009)

The following enhancements were introduced in this release:

- Command Flow feature is a new feature that captures all CICS, DB2, IMS, and IBM MQ commands in chronological order.
- Enhanced Natural and ADABAS support enables the capture of Natural program calls and ADABAS calls in the Natural environment.
- The CICS IA Explorer is now shipped as the CICS IA plug-in for the CICS Explorer.
- DB2 batch jobs for the dependency and command flow data use LOAD and UNLOAD utilities to improve performance.
- Initial support for CSV files: Sample jobs to unload the dependency and command flow data to CSV files rather than to DB2 on z/OS.
- The configuration exec was enhanced to include more configurable options and to store multiple configurations.
- The collector exits were reworked to improve performance.
- The CINQ transaction were removed.
- The load module scanner now recognizes EGL segments.
- You can now capture both Dependency data and Affinity data at the same time.
- Run time options can be refreshed without restarting the collector.
- More CICS and API and SPI commands collected.
- CINT operation and administration changes are logged to the CIULOG.

Enhancements in CICS IA Version 2 Release 2 (2007)

The following enhancements were introduced in this release:

- Support for IBM CICS Transaction Server, Version 3.2
- Queries to identify threadsafe programs and candidates for refactoring as web services
- Support for Software AG Natural
- Capture of secondary resource information
- Intuitive access to CICS relationship data
- Enhanced database schema that captures detailed information for six primary resource types: Transactions, programs, files, temporary storage queues, transient data queues, and web services

Enhancements in CICS IA Version 2 Release 1 (2005)

The following enhancements were introduced in this release:

- A new Eclipse-based GUI enables easy access to the resource relationship data in the database and improved query management facilities. This feature was based on an XML API, which was available to the user. Now replaced by the CICS IA plug-in to the CICS Explorer.
- Timer-based collection is introduced, with improvements to the CINT transaction, which allow the user to control when and in which CICS region the data collection is enabled.
- A program/transaction exclude list is now available.
- ISPF customization of installation jobs is added, making it easier to set up CICS IA for your environment.
- A flag was provided on EXEC CICS START to show if a REQID is present.
- CICS System Definition (CSD) and group-list information is now collected.

Enhancements in CICS IA Version 1 Release 3 (2004)

The following enhancements were introduced in this release:

- Affinity data captured by the Transaction Affinities utility in CICS can be loaded into DB2 tables for analysis.
- Improvements in the data collected for DB2 subsystem resources.
- Length of resource names increased from 50 to 200 bytes, to accommodate ENQ/DEQ names.
- Sample SQL queries that allow resource comparisons on the data in DB2 tables.
- New procedures, including sample data, for the installation process.
- Task Control Block data is collected to assist in assessing threadsafe aspects of CICS DB2 programs.
- Main, auxiliary, and Coupling Facility temporary storage queues are differentiated.
- Sample SQL queries to allow housekeeping functions on the DB2 data.

Enhancements in CICS IA Version 1 Release 2 (2003)

The following enhancements were introduced in this release:

- Support for CICS TS V2.2 commands supporting long TS Queue names, FEPI, TCP/IP, web resources
- Support for remote SYSID information even where not specified in the EXEC CICS command

- Inclusion of calls to DB2, IMS, and IBM MQ remote resources
- The ability to hold information about several CICS regions in one shared VSAM file
- Provision of a single point of control at one terminal for configuring CICS IA options for every region
- The ability to view the last date on which a CICS resource was used

1.4 CICS IA architecture and components

In this section, we provide a high-level overview of the CICS IA components. Later in this book, we provide detailed discussions about the components.

1.4.1 CICS IA components

The design of CICS IA centers on the concept of examining the EXEC CICS commands that the applications and systems programmers use. Each command and its parameters indicate the resources that the program uses. An analysis of these calls provides a view of resource interdependencies. CICS IA also captures resource affinity information.

Figure 1-1 on page 14 shows the components of CICS IA collector.

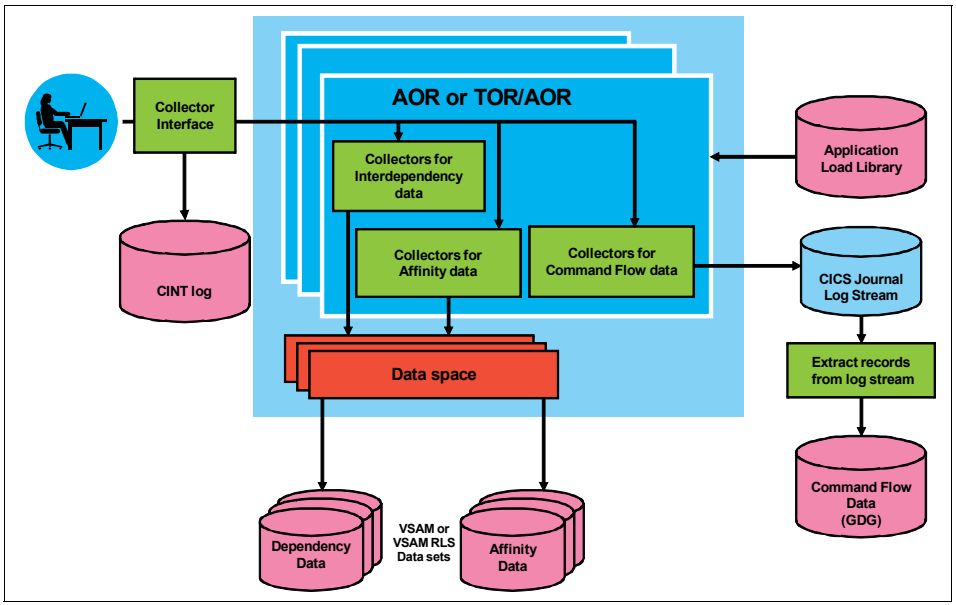


Figure 1-1 CICS IA collector architecture

Figure 1-2 shows the components of CICS IA database architecture.

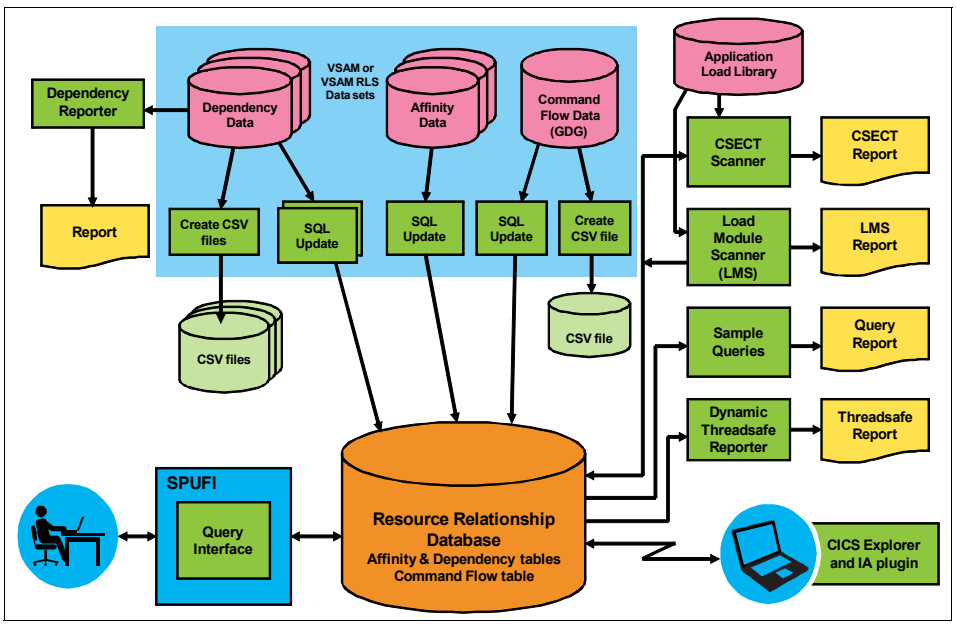


Figure 1-2 CICS IA database architecture

The collector

The collector is now made up of three different components:

- The Dependency Collector
- The Affinity Collector
- The Command Flow Collector

These features were described earlier in the book. See the following references:

- “The Dependency Collector” on page 5
- “The Affinity Collector” on page 6
- “The Command Flow Collector” on page 6

1.4.2 The Load Module Scanner

By using the *Scanner* component of CICS IA, it is possible to write a program to examine the program load modules and report on the EXEC CICS commands and their parameters. The Scanner component produces a report that tells the programming language used and the resources that are involved for each program that the commands issued. You can now load the Scanner information into the DB2 tables CIU.SCAN.SUMMARY and CIU.SCAN.DETAIL.

For more information about running the scanner, see 5.2, “Running the load module scanner” on page 124.

1.4.3 The CSECT Scanner

CICS IA also provides a *CSECT Scanner* that also scans the load modules for information that you can use to identify the version of each CSECT.

The output is stored in DB2 tables, which you can use with the DB2 dependency tables, to identify different versions of programs.

1.4.4 The Dependency Reporter

At specified intervals or on operator command, the data space is written to VSAM files. The *Reporter* component is a set of batch programs that can produce reports from these files. You can run a summary report or a detailed report. The Reporter component will no longer be updated to support new CICS resources. All future development is concentrated on maintaining the DB2 tables. We do not cover the Reporter in this book. For more information about the Dependency Reporter, see the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

1.4.5 The Affinity Reporter and Builder

The Affinity Batch Reporter and Builder allow you to report on your CICS region affinities by affinity type. In order to run an Affinity report, you need to collect Affinity data using the Affinity collector. We do not cover the batch Affinity Reporter and Builder in this book. For more information about the Affinity Reporter and Builder, refer to the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

In CICS IA V5.2, we added the capability to run Affinity Reporter and Builder from the plug-in. This capability is covered in Chapter 4, “Using CICS IA data for affinity analysis” on page 81.

1.4.6 The Threadsafe Reporter

The *Threadsafe Reporter* consists of a batch job that produces reports that display the threadsafe status of each command in the requested programs, specified intervals, or on operator command. The data space is written to VSAM files.

The threadsafe report consists of a header page and one or more pages of program data. The header page lists the report options that are used to create the report and provides definitions for some of the terms that are used in the report. The remaining pages report on each program that meets the criteria that the report options PROGRAMNAME and REGIONNAME specify.

We cover the Threadsafe Reporter in detail in 5.3, “Running the threadsafe report in the plug-in” on page 127.

1.4.7 The CICS IA Explorer plug-in

The CICS IA Explorer perspective and connections are shipped as a plug-in into the CICS Explorer client, an Eclipse Rich Client platform and Java Runtime Environment. It connects to the CICS IA DB2 relationship repository on IBM Systems z using JDBC Type 4 drivers. The IA plug-in has its own connection to the DB2 database. More information about the connection can be found in the plug-in help for the Explorer. To obtain help, select **Help → Help Contents** and select the CICS IA plug-in guide. Then select “Configuring the CICS IA connection” as shown in Figure 1-3 on page 17.

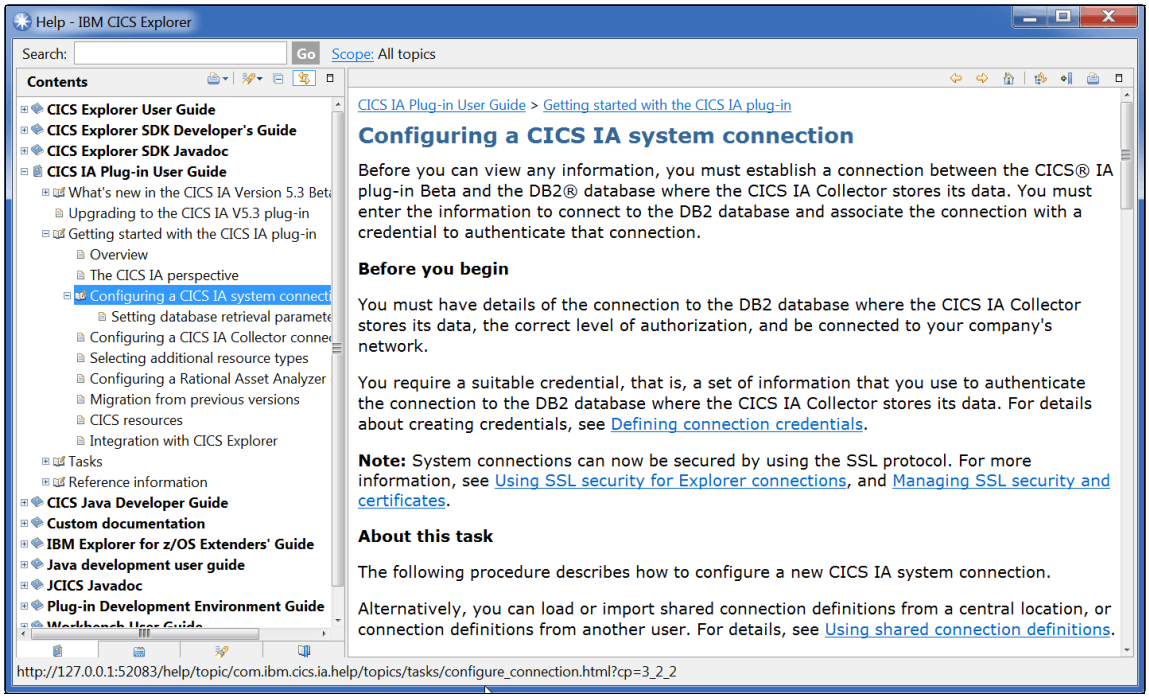


Figure 1-3 IA Explorer plug-in help

The CICS IA V5.3 Explorer plug-in has been reworked to present a new and easier to use perspective. The perspective is shown in Figure 1-4.

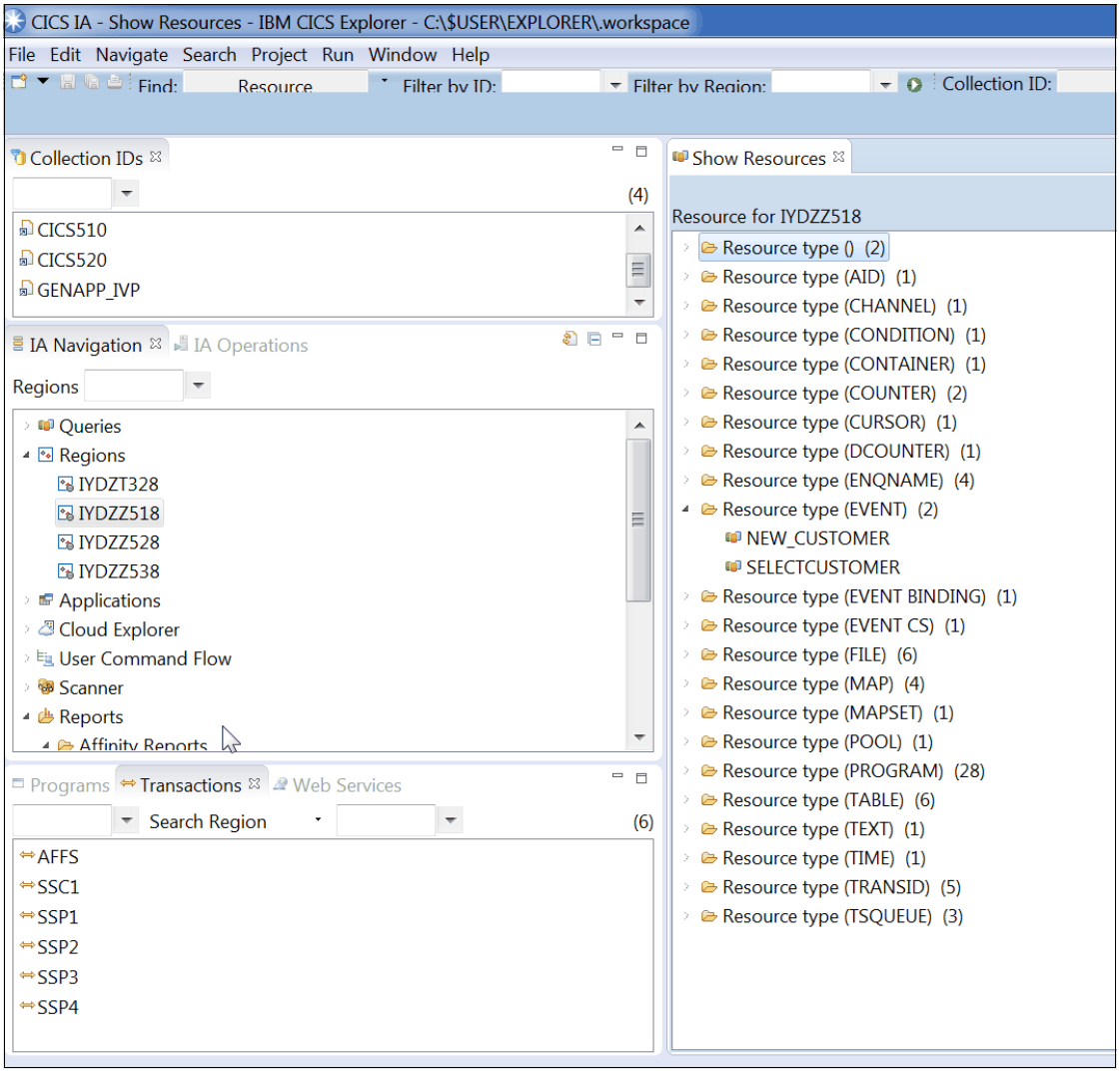


Figure 1-4 The new IA V5.3 perspective

The left side of the perspective has now been reduced to six navigational views:

- Collection IDs
- IA Navigation
- IA Operations
- Transactions

- Programs
- Webservices

Collection IDs view

You can use this view to set the scope of your dependency queries. A CICS IA collection ID is added to the dependency tables when you load the data into DB2. For more information about collection IDs, see 2.3, “Using the collection ID to manage your data” on page 44.

IA Navigational view

This view is new in the IA V5.3 perspective and contains a navigational tree of folders that were previously views in themselves, as shown in Figure 1-5.

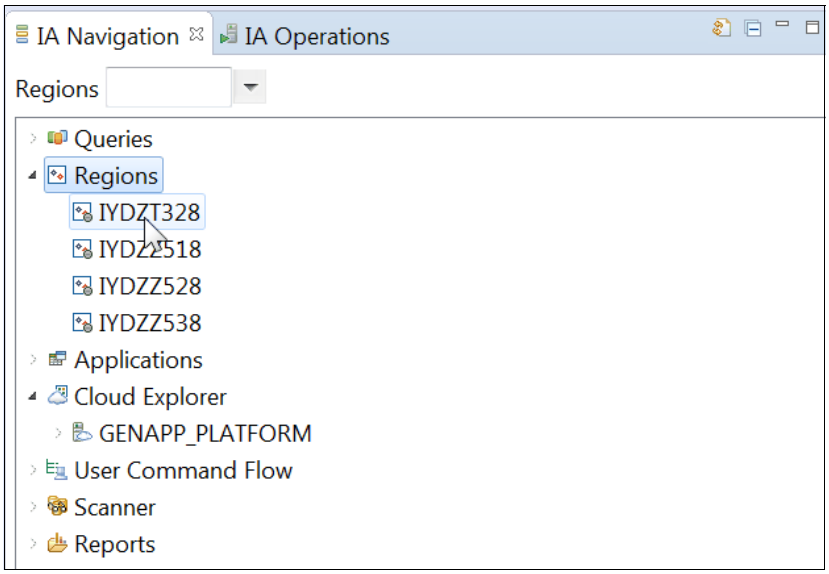


Figure 1-5 New IA Navigational view

From this IA Navigational view, you can do the following actions and much more:

- Run queries
- Run reports
- Look at Command Flow output
- Visualize IA data by Application and Platform
- View your reports
- Look at resources by region

This view is used extensively when we look at the different scenarios described in this book.

IA Operations view

This view was updated in CICS IA V5.3 to include scenario-based collections. The CICS IA host component provides Webservices to manage this view and it therefore has its own connection other than the JDBC connection to the database. Use “Help Content” to review how to set up this connection. This view allows you to operate both Dependency and Affinity collection by CICS regions as shown in Figure 1-6.

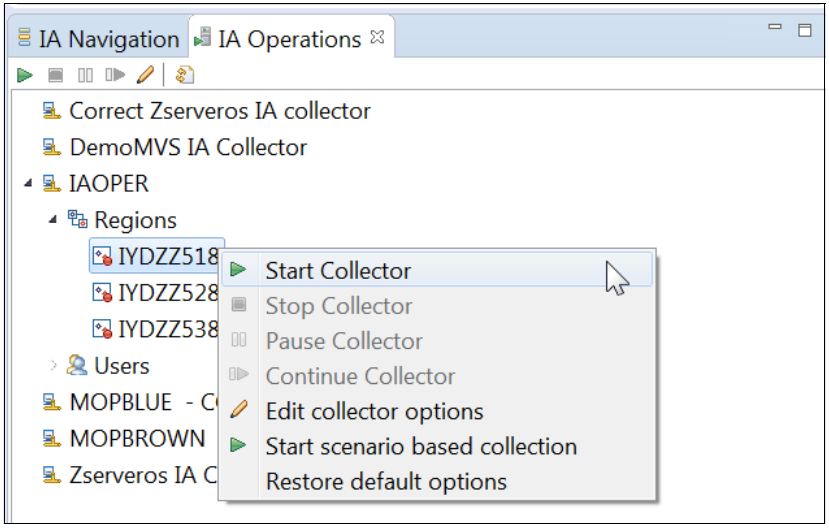


Figure 1-6 IA operations against a CICS region

You can also administer and operate the command flow collection from this view as shown in Figure 1-7 on page 21.

Transaction, Program, and Webservices views

These views provide a list of transactions, programs, or Webservices, respectively. From these views, you can do many things as shown in Figure 1-8 on page 21.

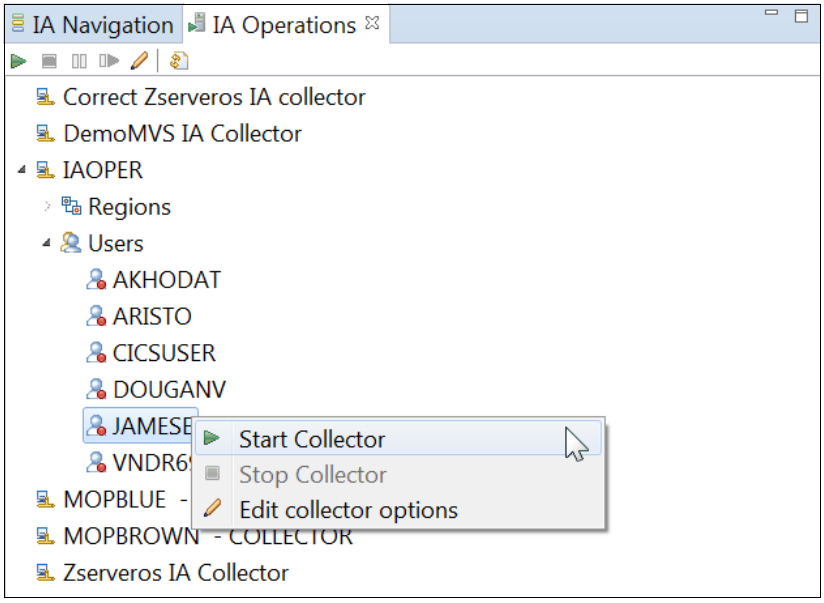


Figure 1-7 Managing the command flow collector for user JAMESE

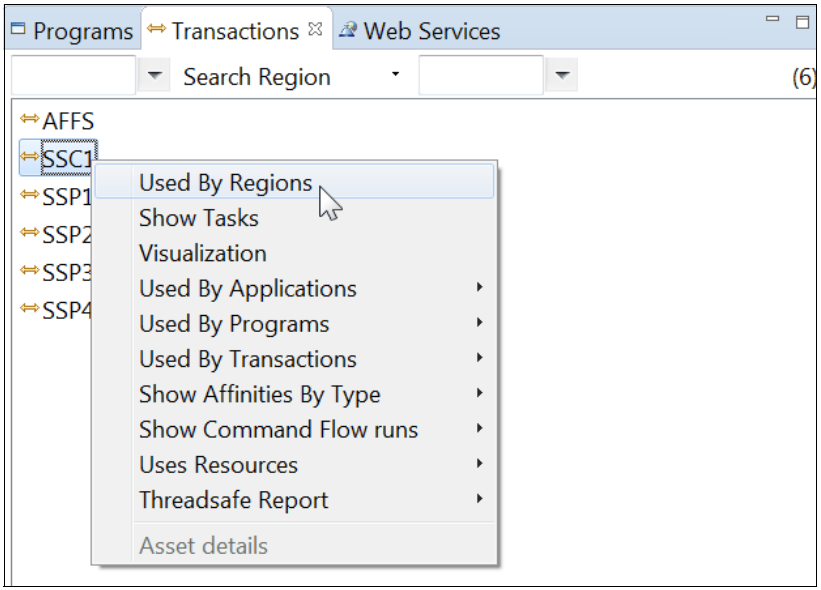


Figure 1-8 Commands available against a transaction

In the IA V5.3 perspective, the results of issuing queries or reports from the left navigational views all result in a new view being opened on the right side for each type of query. The example in Figure 1-9 shows the “results” views for the following tabs:

- Show Resources query
- Uses resources
- Resource Visualization
- Used By
- Command Flow

The Command Flow view is on top. You can now easily move from one results folder to the next.

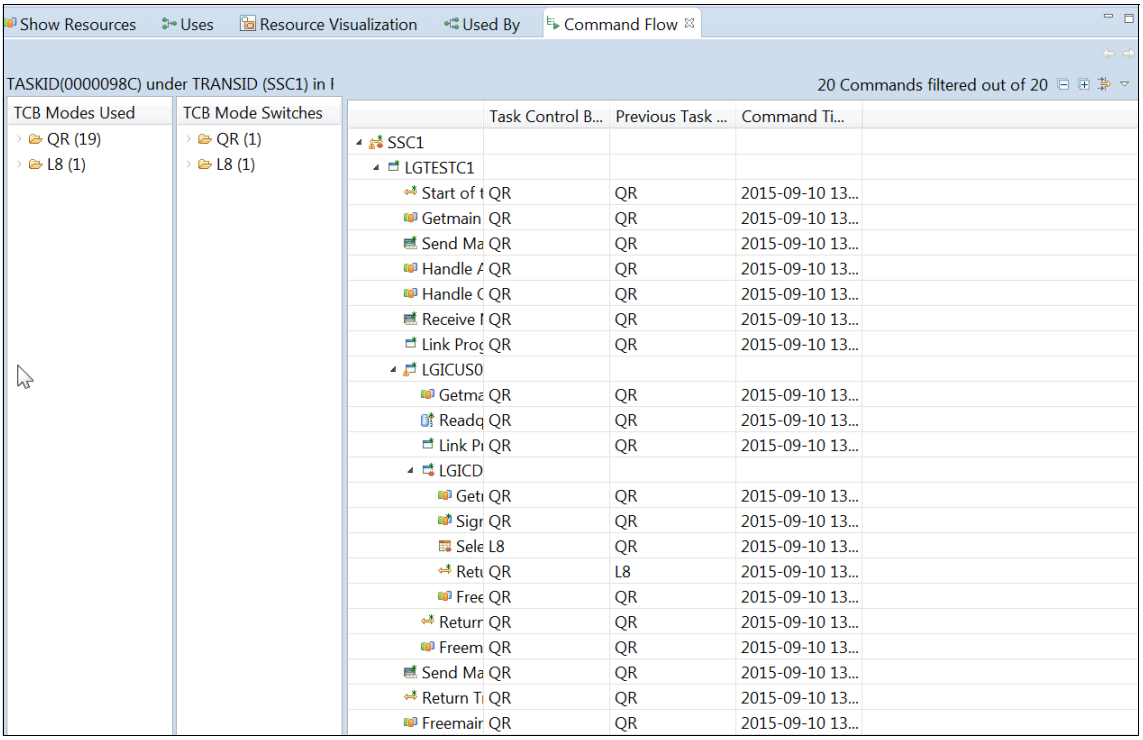


Figure 1-9 The right side contains all the “results” views

The “IA plug-in” is used extensively in the rest of this book.

2

Preferred practices

In this chapter, we provide information about CICS Interdependency Analyzer (IA) preferred practices to consider before you use CICS IA.

We look at how to configure CICS IA to do the following actions:

2.1 Collecting CICS IA data in a production region

CICS IA dependency and affinity data is analyzed to help you make decisions in order to reduce the cost, risk, and impact of changes to your application and the environment on which it runs. The quality of these decisions is directly related to the quality of the data. Many customers feel that collecting data in their production regions gives them the best data in order to gain an understanding of the relationships and dependencies of their CICS applications and environment.

In this section, we look at the CICS IA settings that allow you to run CICS IA collections in a production region with the minimum impact to the performance of your applications.

There are many ways to control the performance of the CICS IA collection. In this section, we look at the following options:

- Using the Optimum Collection option
- IA threadsafe considerations
- IA exclude lists
- IA resource compression and prefix list
- Dynamic COBOL calls

2.1.1 Using the Optimum Collection option

In CICS IA V5.2, we introduced two new options to the “General Options” administration panel, which is available via the CINT transaction. The following options are available:

- Trigger for Task Collection
- Collect Long Running Tasks

These two new options apply to the collection of dependency data only. They are not used during the collection of “Affinity” data. The collection of “Affinity” data in a production region is covered in 4.2.1, “Configuring affinity options” on page 94.

The “Trigger for Task Collection” option allows you to specify on how many CICS tasks you want to collect data for. The value for this option is a numeric in the range 1 - 9999. If the value is set to “n,” CICS IA collects dependency data for every “n”th task that is started. For example, if you set the value to 10, CICS IA collects dependency data on every 10th task. To collect data for all tasks, set the value to 1.

The performance impact of the CICS IA collector exits on your application is governed by how much machine instruction is executed by the exits. By reducing the number of tasks that you collect data for, you reduce the impact of each exit invocation. If the “Trigger for Task Collection” is set to 10, the collector exits will still be called for each task that is started, but will be fully processed for the 10th task when the data is written to the data space. For the other nine tasks, the exits are invoked, the task number checked, and the exit terminates and returns control to the application.

So what is the “optimum” value for this field?

The CICS performance team in Hursley carried out some tests and their optimum value for this field was 50. Setting a value higher than 50 did not show any significant increase in performance savings. The results of the tests can be seen in Appendix B, “Task collection frequency: Performance results” on page 235.

2.1.2 CICS IA threadsafe consideration

The CICS IA collection exits are driven by your application invoking a CICS command, DB2 command, IMS command, or an IBM MQ command. These exits can run on both the QR TCB and open TCBs and therefore must be threadsafe. All of the exits update the CICS IA dataspace. To make the exits threadsafe, the access to the data space must be serialized. This is done by using the CICS ENQUEUE and DEQUEUE technique.

As well as your applications accessing the data space, our long running task, CINB, also needs access to the data space to enable the offload to the VSAM files. Therefore, we use the ENQUEUE and DEQUEUE technique in the CINB transaction as well. This is illustrated in Figure 2-1 on page 26.

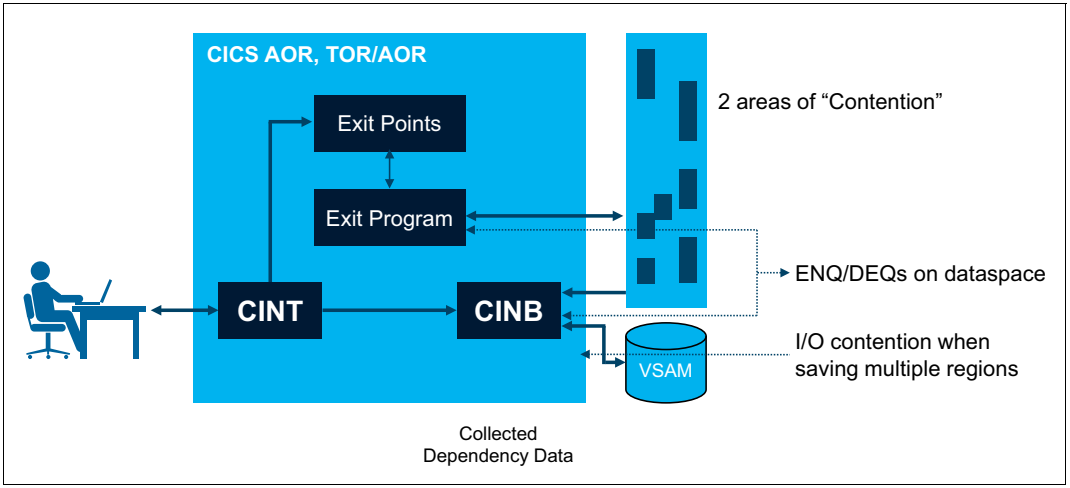


Figure 2-1 CICS IA access to the data space

The long running task CINB can be woken up to start a save to VSAM in two ways:

- Perform Periodic saves
- Trigger for CINB starts

The “Perform Periodic saves” option causes the CINB transaction to wake up every 5 minutes and start a save.

The “Trigger for CINB starts” option causes the CINB transaction to wake up every time a preconfigured number of records is updated.

Having CINB wake-up during collection can lead to the following contentions as shown in Figure 2-1:

- Increased ENQ/DEQ contention on the IA data space
- IO contention on several CICS regions writing out to the VSAM files

Therefore, the advice is to switch off the CINB save process during the collection and only save to VSAM during a PAUSE or STOP of the collector.

You can switch off both periodic saves and triggered saves by selecting the following general options using the CINT transaction:

- Perform periodic saves: N (Y=Yes, N=No)
- Trigger for CINB start: 1 (2 - 9999 thousand record updates)

For more information about the CINT transaction, see the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html

When the CINB save function is switched off, the only ENQ/DEQs are caused by the applications driving the CICS IA exits and therefore the contention is greatly reduced.

To force a daily save of the data from the data space to the CICS IA VSAM files, use option 2, “Time and Date” options as shown in Figure 2-2.

The default time and date options are shown in Figure 2-3 on page 28. This is set to collect all day, every day, every week, and so on.

```
CIU29O        CICS Interdependency Analyzer for z/OS - V5R3M0       2015/10/12
                     Resource Options for                           01:26:34PM
               CICS Sysid:  DFTS  CICS Applid:  DEFAULTS

Type action code then press ENTER: 2

    1 = General Options
    2 = Time/Date Options

    Interdependency Options            Affinity Options
    3 = CICS Options for APIs          7 = Affinity Options
    4 = CICS Options for SPIs
    5 = DB2/IMS/MQ/CPSM Options        CICS TS Applications
    6 = Natural Options                8 = Application Data




CICS Sysid: Z538    CICS Applid: IYDZZ538    TermID: TC78
CIU7029I Action processed successfully.
F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Exit
```

Figure 2-2 Select option 2 for time and date options

```
CIU280        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/12
                      Time and Date Options for                      01:31:10PM
                  CICS Sysid: DFTS     CICS Applid: DEFAULTS

Modify the options and press Enter to update or F12 to Cancel.
Hour slots: Y=Yes, N=No, A=Affinity, I=Dependency, B=Both
Month, Day, and Week slots: Y=Yes, N=No

Hour of day: 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24
             Y Y Y Y Y Y Y Y Y Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y

Day of week: Mon  Tue  Wed  Thu  Fri  Sat  Sun
             Y    Y    Y    Y    Y    Y    Y

Day of month: 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 20 1 2 3 4 5 6 7 8 9 30 1
              Y Y Y Y Y Y Y Y Y Y  Y Y Y Y Y Y Y Y Y Y  Y Y Y Y Y Y Y Y Y Y  Y

Month of year: Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
               Y    Y    Y    Y    Y    Y    Y    Y    Y    Y    Y    Y

CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC78

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 2-3 Default time and date options

You can override the default options for each CICS region being collected to stop the collector for an hour every day and thus save your data to VSAM file. In Figure 2-4 on page 29, we selected to choose to switch off the collector for region IYDZZ528 for 1 hour from midnight. This causes the data in the data space to be written out to VSAM. You can stagger the writing of data to the VSAM files from your shared collection regions by selecting a different hour for each CICS region and override the default.

```
CIU280        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/12
                      Time and Date Options for                      01:34:46PM
                  CICS Sysid: Z528     CICS Applid: IYDZZ528

Modify the options and press Enter to update or F12 to Cancel.
Hour slots: Y=Yes, N=No, A=Affinity, I=Dependency, B=Both , or BLANK
Month, Day, and Week slots: Y=Yes, N=No , or BLANK

Hour of day: 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24
             N

Day of week: Mon  Tue  Wed  Thu  Fri  Sat  Sun


Day of month: 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 20 1 2 3 4 5 6 7 8 9 30 1


Month of year: Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec


CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC78

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 2-4 Switch off the collector for region IYDZZ528

By using these options, you can manage both the ENQ/DEQ contention and the IO contention.

2.1.3 Using the IA exclude lists

CICS IA provides a number of exclude lists that can be used to reduce the performance impact of running both the dependency collector and the command flow collector:

Transaction exclude list You can use this list to exclude transactions from collection.

Program exclude list You can use this list to exclude programs from collection.

Command exclude list You can use this list to exclude certain CICS commands from collection. This list was added in CICS IA V5.3.

All three exclude lists come in the form of an assembler DSECT that can be assembled and linked. Sample source and sample JCL are supplied by CICS IA.

The linked load modules are loaded into storage when the collectors are started. The exclude lists that are loaded are configurable using the “General Options” using the CINT transaction. Following are the options:

- Program exclude list: CIUXPROG (1 - 8 characters)
- Transaction exclude list: CIUXTRAN (1 - 8 characters)
- Command exclude list: CIUXCOMM (1 - 8 characters)

Similar options are provided with the CINC Command Flow transaction.

You can take these samples and modify them to suit your environment as the default exclude lists for all your CICS regions or you can take a copy of the lists and create region-specific lists. We create a region-specific list for the transaction exclude list in the next section.

Transaction exclude list

The sample source for the transaction exclude list can be found in:

<hlq>.SCIUSRCE(CIUXTRAN)

The sample JCL to assemble and link CIUXTRAN can be found in:

<hlq>.SCIUSAMP(CIUJCLXT)

We now show the steps required to exclude all transactions starting SSC from collection using a transaction exclude list called CIUXTSSC and then configure a CICS region to use this exclude list rather than the default list.

First, we need to take a copy of the shipped source shown in Figure 2-5 and edit it to add transaction prefix SSC and change the DSECT name to CIUXTSSC as shown in Figure 2-6 on page 31.

```
CIUXTRAN CSECT
CIUXTRAN AMODE 31
CIUXTRAN RMODE ANY
         DS    0F
* Add user prefixes here
*        DC    AL1(3),C'TST'          Example
         DC    AL1(0)                 End of list
         END   CIUXTRAN
```

Figure 2-5 Sample transaction exclude list: This exclude list is empty

```
CIUXSSCT CSECT
CIUXSSCT AMODE 31
CIUXSSCT RMODE ANY
         DS    0F
* Add user prefixes here
         DC    AL1(3),C'SCC'          All transactions starting SSC
         DC    AL1(0)                 End of list
         END   CIUXSSCT
```

Figure 2-6 Transaction exclude list for all SSC transactions

We then need to assemble and link the source to create the exclude list load module. Again, use the sample JCL job CIUJCLXT and modify as shown in Figure 2-7.

```
//ASM      EXEC PGM=ASMA90,
//             PARM='OBJECT,LIST,ALIGN,TERM'
//SYSIN    DD DISP=SHR,
//    DSN=ANTZ.CICS.IA.DEV.EMSINT.SCIUSRCE(CIUXTSSC)
//SYSPRINT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(5,5))
//SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(5,5))
//SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(5,5))
//SYSLIN   DD DSN=&&OBJ,DISP=(,PASS),UNIT=SYSDA,
//            SPACE=(TRK,(15,5))
//*        LINK-EDIT MODULE
//LKED     EXEC PGM=IEWL,COND=(4,LT),
//         PARM='RENT,LIST,LET,MAP,NCAL'
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(2,1))
//SYSLMOD  DD DISP=SHR,
//   DSN=ANTZ.CICS.IA.DEV.EMSINT.SCIULOAD(CIUXTSSC)
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=&&OBJ,DISP=(OLD,DELETE)
```

Figure 2-7 Sample job to assemble and link CIUXTSSC

You now need to add a CICS resource definition for this program in the DFHCSD for the CICS regions that use this exclude list as shown in Figure 2-8. If you are using program auto-install, skip this step.

```
DEFINE PROGRAM(CIUXTSCC)      GROUP(CINTGR53)
       DESCRIPTION(TRANSACTION EXCLUDE LIST for SCC)
       LANGUAGE(ASSEMBLER)
       RELOAD(NO)             RESIDENT(NO)
       USAGE(NORMAL)          USELPACOPY(NO)
       STATUS(ENABLED)        CEDF(NO)
       DATALOCATION(ANY)      EXECKEY(CICS)
```

Figure 2-8 Program definition for CIUXTSCC

Lastly, override the default value for the CICS region in which you want to use this exclude list. We use our new list, CIUXTSCC, in CICS region IYDZZ528. Using the CINT option 2 for “Configure Region options,” select option 4 for “options” against region IYDZZ528 as shown in Figure 2-9.

```
CIU200        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/13
                        Region Configuration Menu                    12:04:41PM

Type action code then press ENTER.                                   More :
1=Add Region   2=Copy Region   3=Delete Region   4=Options


      CICS      CICS   New       New
 Act  Applid    Sysid  Applid    Sysid  Status       Collecting
      DEFAULTS  DFTS
      ALL       ALL
      IYDZZ518  Z518                   STOPPED
  4   IYDZZ528  Z528                   STOPPED
      IYDZZ538  Z538                   STOPPED









CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC30

F1=Help      F2=           F3=Exit      F4=          F5= Refresh  F6=
F7=Page Up   F8=Page Down  F9=          F10=         F11=         F12=
```

Figure 2-9 Configure the options for region IYDZZ528

You then select 1 for “General Options” and enter CIUXTSSC for the transaction exclude list as shown in Figure 2-10 on page 33.

```
CIU26O        CICS Interdependency Analyzer for z/OS - V5R3MO        2015/10/13
                          General Options for                        12:08:19PM
          CICS Sysid   : Z528     CICS Applid   : IYDZZ528
Modify the options and press Enter to update, or PF12 to Cancel.
Control options (Fields may be set to blanks for default)
  Data to Collect  . . . . . . :      (A=Affinity, I=Interdependency, B=Both)
  Perform periodic saves . . . :      (Y=Yes, N=No)
  Trigger for CINB start . . . :      (2 to 9999 thousand record updates)
  Restore data on start  . . . :      (Y=Yes, N=No)
  Maintain usage counts  . . . :      (Y=Yes, N=No)
  Size of dataspace  . . . . . :      (10 to 2000 Mbytes)
  Transid prefix (optional). . :      (1 to 4 characters)
  Program exclude list . . . . :          (1 to 8 characters)
  Transaction exclude list . . : CIUXTSCC (1 to 8 characters)
  Command exclude list . . . . :          (1 to 8 characters)
  Resource compression list. . :          (1 to 8 characters)
  Dump HLQ . . . . . . . . . . :          (1 to 8 characters)
  Dynamic call . . . . . . . . :      (Y=Yes, N=No)
  Trigger for Task collection. :      (1 to 9999)
  Collect Long Running Tasks . :      (Y=Yes, N=No)
CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC30

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 2-10 Set the transaction exclude list to CIUXTSCC

Program exclude list

The sample source for the program exclude list can be found in:

`<hlq>.SCIUSRCE(CIUXPROG)`

The sample JCL to assemble and link CIUXPROG can be found in:

`<hlq>.SCIUSAMP(CIUJCLXP)`

The sample for the program exclude list contains program prefixes for some of the IBM licensed programs as shown in Figure 2-11 on page 34.

Again, you can update the default table CIUXPROG or create region-specific exclude lists as we did for the transaction exclude list.

```
CIUXPROG CSECT
CIUXPROG AMODE 31
CIUXPROG RMODE ANY
         DS    0F
* Add user prefixes here
*        DC    AL1(4),C'TEST'         Example
* Predefined prefixes
         DC    AL1(3),C'DFH'          CICS
         DC    AL1(3),C'CEE'
         DC    AL1(3),C'EQA'
         DC    AL1(3),C'IBM'
         DC    AL1(3),C'EDC'
         DC    AL1(3),C'IGZ'
         DC    AL1(3),C'CAU'          Affinities utility
         DC    AL1(3),C'CIU'          Interdependency utility
         DC    AL1(3),C'DSN'          DB2
         DC    AL1(3),C'EYU'          CICSPlex SM
         DC    AL1(3),C'CJA'          CICSPlex SM for CICS640
         DC    AL1(3),C'CJB'          CICSPlex SM for CICS650
         DC    AL1(3),C'CJC'          CICSPlex SM for CICS660
         DC    AL1(3),C'CJD'          CICSPlex SM for CICS670
         DC    AL1(3),C'CJE'          CICSPlex SM for CICS680
         DC    AL1(3),C'CSQ'          MQ
         DC    AL1(3),C'CMZ'          CICS PM
         DC    AL1(3),C'CPA'          CICS PA
         DC    AL1(3),C'ABL'          OTTO
         DC    AL1(3),C'CBM'          CICS BEP
         DC    AL1(3),C'DWW'          CICS VR
         DC    AL1(3),C'ISZ'          Session manager
         DC    AL1(3),C'VID'          VT
         DC    AL1(3),C'CBK'          CICS BAC
         DC    AL1(3),C'CCV'          CICS CM              @R57780C
         DC    AL1(3),C'CTG'          CICS TRANSACTION GATEWAY
         DC    AL1(3),C'CPH'          CICS DA
         DC    AL1(4),C'IN25'
         DC    AL1(8),C'ADABAS  '     Natural ...                @R97019A
         DC    AL1(8),C'ADACICT '      related                   @R97019A
         DC    AL1(0)                 End of list
         END   CIUXPROG
```

Figure 2-11 Sample program exclude list

Command exclude list

The sample source for the command exclude list can be found in:

`<hlq>.SCIUSRCE(CIUXCOMM)`

The sample JCL to assemble and link CIUXCOMM can be found in:

`<hlq>.SCIUSAMP(CIUJCLXC)`

This list was added in CICS IA V5.3. It was added to provide the ability to have more control on the actual CICS commands we collect. The “CICS options for APIs” and the “CICS options for SPIs” in the CINT transaction provides you with some degree of control on the CICS commands you want to collect. This list gives you further control by allowing you to specify the EIBFN code for the command you want to exclude. You can also exclude by command groups.

The example in Figure 2-12 excludes all the TDQUEUE commands and the **WRITE JOURNAL** command.

```
EJECT
CIUXCOMM CSECT
CIUXCOMM AMODE 31
CIUXCOMM RMODE ANY
         DS    0F
* Add command and group code here
         DC    AL1(2),X'0800'      Exclude TDQ commands
         DC    AL1(2),X'1402'      Exclude write journal command
         DC    AL1(0)                 End of list
         END   CIUXCOMM
```

Figure 2-12 Example command exclude list

Again, you can update the default table CIUXCOMM or create region-specific exclude lists as we did for the transaction exclude list.

2.1.4 Using the IA resource compression list

In CICS IA V5.3, we provide a resource compression list. This list replaces the resource prefix list provided by earlier releases of CICS IA.

The purpose of the resource compression list is to compress records in CICS IA dataspace to reduce memory usage, avoid output of unnecessary data into DB2 tables, and improve performance. In CICS IA V5.3, we now have three formats of the resource list. The three formats are not compatible and cannot be used in the same DSECT.

The resource compression is performed against the following commands:

- IBM MQ commands
- CHANNELs/CONTAINERs commands
- ENQ/DEQ commands
- TS Queue commands
- CANCEL/POST/DELAY commands

Records that match rules specified in this resource list are collected as just one record. If a match is found, the rest of the resource name is changed to contain + (plus) signs.

Version 1 format

The format for this rule is as follows:

- The prefix length (range 1 - 32)
- The resource name prefix, which is a string of valid symbols

For example, if we have a TSQUEUE name that is made up of a PREFIX “DEPT10” followed by a numeric field as shown below:

DEPT10nnnnnn

Without compression, we would write a record in the dataspace and subsequently a DB2 row for all instances. If we had WRITEQ, READQ, and DELETEQ for this resource name, we could see up to 3,000,000 records:

100,000 records for each of the three commands starting with DEPT10000000 through to DEPT10999999

By using the compression rule shown in Example 2-1, we can reduce the number of records to three:

One record for each of the three commands named DEPT10++++++

Note: Any of the supported resources starting with the name DEPT10 will also be replaced.

Example 2-1 Format 1 of the compression rule

```
CIUXRCOM CSECT
CIUXRCOM AMODE 31
CIUXRCOM RMODE ANY
         DS    0F
* Add your own rules here.
         DC    AL1(6),C'DEPT10'
         DC    AL1(0)                    End of list
        END   CIUXRCOM
* End of the list:
```

Version 2 format

The format for this rule is as follows:

- The command types to be compressed
- Starting position of the key (range 1 - 32)
- The key length (range 1 - 32)
- The key that is a string, which can be a prefix, an infix, or a suffix

The following command types are supported:

- decimal 1 = MQ
- decimal 4 = CHANNELs/CONTAINERs
- decimal 7 = ENQ/DEQ
- decimal 10 = TS Queue
- decimal 11 = CANCEL/POST/DELAY

In this example, we have a REQID compromising of a numeric followed by ABCD as shown in nnnnABCD.

This REQID can be associated with the **CANCEL**, **POST**, and **DELAY** command.

Again, without a compression rule we could have many records in the dataspace and in the DB2 tables.

By using the compression rule shown in Example 2-2, we can reduce the number of records to 3:

One record for each of the three commands named ++++ABCD

Example 2-2 Format 2 of the compression rule

```
CIUXRCOM CSECT
CIUXRCOM AMODE 31
CIUXRCOM RMODE ANY
         DS    0F
* Add your own rules here.
         DC    AL1(11),AL1(4),AL1(4),C'ABCD'   CANCEL/POST/DELAY
         DC    AL1(0)                    End of list
           END  CIUXRCOM
* End of the list:
```

Version 3 format

The format for this rule is as follows:

- The command types to be compressed
- Compression trigger (range 1 - 255)
- Starting position of the save area (range 1 - 32)
- The save area length (range 0 - 32)
- Starting position of the key (range 1 - 32)
- The key length (range 0 - 32)
- Starting position of the key (range 1 - 32)
- The key that is a string, which can be a prefix, an infix, or a suffix

The following command types are supported:

- decimal 1 = MQ
- decimal 4 = CHANNELs/CONTAINERs
- decimal 7 = ENQ/DEQ
- decimal 10 = TS Queue
- decimal 11 = CANCEL/POST/DELAY

The save area defines a part of the resource name that will be preserved (not filled with +). It may include the key.

The compression trigger is the number of records at which the compression rule is applied and compression is started.

Before adding rules, first specify the eyecatcher to let CICS IA know that version 3 of resource compression list is used:

```
DC    CL8'PLIST V3'
```

In this example, we compress TSQUEUE commands using a resource name of TSQnnnnn where “nnnnnn” increments form 0 - 999999. We also distinguish between TSQ1nnnn, TSQ2nnnn, and TSQ3nnnn, and so on, and only start the compression after we have seen two instances for TSQ1, TSQ2, and so on.

Example 2-3 Format 3 of the compression rule

```
CIUXRCOM CSECT
CIUXRCOM AMODE 31
CIUXRCOM RMODE ANY
         DS    0F
* Add your own rules here.
         DC    CL8'PLIST V3'
         DC    AL1(10),AL1(3),AL1(4),AL1(1),AL1(1),AL1(3),C'TSQ'
         DC    AL1(0)                    End of list
        END   CIUXRCOM
* End of the list:
```

By applying the rule in Example 2-3, we get the following resource names associated with the TSQUEUE commands:

```
TSQ00001

TSQ00002

TSQ0++++ compression starts after seeing more than two “TSQ0****”
entries

TSQ10000

TSQ10001

TSQ1++++ compression starts after seeing more than two “TSQ1****”
entries

TSQ20000

TSQ20001

TSQ2++++ compression starts after seeing more than two “TSQ2****”
entries

...

TSQ90000

TSQ90001

TSQ9++++ compression starts after seeing more than two “TSQ1****”
entries
```

Again, you can update the default table CIUXRCOM or create region-specific compression lists as we did for the “Transaction exclude list” on page 30.

2.1.5 Dynamic COBOL calls

The CICS IA collector can detect dynamic COBOL calls in an IBM Language Environment®. To support the collection of dynamic IBM COBOL Language Environment calls, CICS IA assumes that the Call parameter list for IBM COBOL for OS/390® program conforms to the structure documented in the publication Language Environment Vendor Interfaces for COBOL Call routine. In order to detect dynamic COBOL calls for IBM COBOL for OS/390 and IBM VS COBOL II programs, the CBLPSHPOP Language Environment option must be active. The Language Environment option ALL31(ON/OFF) is supported for dynamic calls from IBM COBOL for OS/390 programs.

For more information about what can and cannot be collected, see the section “Collecting dynamic COBOL calls” in the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

There are two requirements that CICS IA needs to achieve in order to support dynamic COBOL calls:

- Identifying the actual dynamic call, for example, PROGA calls PROGB
- EXEC CICS commands or any RMI commands issued by PROGB are reported under PROGB

The first requirement is met only if the CBLPSHPOP Language Environment option is active.

The second requirement is met by using the CICS SPI call `IDENTIFY_PROGRAM`. This call can be CPU intensive, and with this fact in mind, we added an option to switch the collection of COBOL dynamic calls ON or OFF for both the dependency collector and the command flow collector.

For the dependency collector, go to the “General Options” panel using CINT and update the following option:

```
Dynamic call . . . . . : Y    (Y=Yes, N=No)
```

The same option can be found on the “Command Flow Options” panel using the CINC transaction.

2.2 Collecting the DB2 resource name

CICS IA interfaces with DB2 in two ways:

- It collects DB2 resources and commands used within a CICS region. In this case, we may collect resource information for many different DB2 subsystems depending on which DB2 subsystem the CICS region is connected to.
- It stores the collected data in a DB2 database. Usually, you have one CICS IA database where you can load all your collected data. This database can include data collected from development, test, and production regions. It can also include data from all the DB2 subsystems connected to these CICS regions.

The DB2 resource collection is performed by using the XRMIIN and XRMIOUT CICS exits. When an application issues an **EXEC SQL** command, information is written to the dataspace. An entry is written to the IA dataspace for this command including the “stmtno”, “sequence number” and the “plan name.”

Note: The actual resource name associated with the command is unavailable to us.

The resource name is obtained during the save process from the dataspace to the VSAM file for DB2 data. The resource name is obtained by accessing the SYSIMB.SYSPACKSTMT table or the SYSIBM.SYSSTMT tables. At many customer sites these tables are not maintained and can include millions of rows. In order to improve the performance of the CINB save process for DB2, we supply DB2 indexes for these tables.

The supplied index can be found in the following command:

```
<hlq>.SCIUSQL(CIUIBM1)
```

A sample job to create this index can be found in the following command:

```
<hlq>.SCIUSAMP(CIUDBCT)
```

This job needs to be run for each DB2 subsystem that you intend to collect DB2 data for.

Many customer sites do not allow these indexes to be created. Therefore, we have two options to avoid performance issues during CINB save.

Turn off access to the SYSIBM tables

Use transaction CINT and select 2 for “Configure Region Options.” Then, enter option 4 for the DEFAULT record or a specific region record, as shown in Figure 2-13.

```
CIU200        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/13
                         Region Configuration Menu                   02:44:01PM

Type action code then press ENTER.                                   More :
1=Add Region   2=Copy Region   3=Delete Region   4=Options


      CICS      CICS   New       New
 Act  Applid    Sysid  Applid    Sysid  Status       Collecting
  4   DEFAULTS  DFTS
      ALL       ALL
      IYDZZ518  Z518                   STOPPED
      IYDZZ528  Z528                   STOPPED
      IYDZZ538  Z538                   STOPPED









CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC30
```

Figure 2-13 Select “Options” for the DEFAULT record

Then, select option 5 for “DB2/IMS/MQ/CPSM Options” and switch off the “Collect resource name” flag, as shown in Figure 2-14.

```
CIU250        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/13
               DB2/MQ/IMS/CPSM       Resource Options for            02:47:47PM

          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.
Detect command types: Y=Yes, N=No

DB2 Options
  Collect DB2 Resources . . . . . : Y
  Collect resource name . . . . . : N (Y/N) (Access the SYSIBM.SYSPACKSTMT
                                       and the SYSIBM.SYSSTMT tables)
MQ Options
  Collect MQ Resources  . . . . . : Y

IMS Options
  Collect IMS Resources . . . . . : Y

CPSM Options
  Collect CPSM Resources  . . . . : Y
CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC30
```

Figure 2-14 Switch off “Collect resource name”

Note: By switching off the Collect resource name option, we still capture that the program issued an SQL command, but we have a blank resource name.

Create a copy of the SYSIBM tables

First, you must take a daily copy of the SYSPACKSTMT table overnight using a different qualifier or ALIAS. In our example, we use a DB2 qualifier of “IAQUAL”.

By default, the CINB programs are currently bound to use the SYSIBM qualifier to access the SYSIBM tables. You need to configure the DB2 collector options to use your modified qualifier. To do this modification, run the configuration exec by entering the following TSO command:

```
ex '<iahlq>.SCIUEXEC(ciucnfg1)' '<iahlq> ENU'
```

Then, select option 3 to configure a new CICS collection or 4 to configure an existing CICS collection. In this example, we use an existing collection as shown in Figure 2-15.

```
********** CICS Interdependency Analyzer for z/OS - V5R3M0 **********
Command ===>

Press ENTER to complete, PF3 to go back or PF1 for help.

Please select an option from the list below:


   4  0. Settings
      1. Configure new DB2
      2. Configure existing DB2
      3. Configure new CICS
      4. Configure existing CICS
      5. Configure new UDB
      6. Configure existing UDB
```

Figure 2-15 Select an existing CICS collection

Then, select “f” for a full configuration of IYDZZ528, as shown in Figure 2-16.

```
************** CICS Interdependency Analyzer for z/OS - V5R3 Row 1 to 10 of 10
Command ===>                                                  Scroll ===> CSR

                     CICS IA CICS Customization Function

Press ENTER to complete, PF3 to go back or PF1 for help.

For selecting a CICS configuration from the list below, please type
S for Short Configuration, F for Full Configuration, D for Delete,
C for Copy or R for Rename.

Cmd  Configuration  CICSVersion  Description
     IYDZT524       V520         test
     IYDZZ518       V510         test
     IYDZZ524       V520         test
f    IYDZZ528       V520         test
     IYDZZ529       V520         test
```

Figure 2-16 Select configuration IYDZZ528

Then, scroll down to the DB2 variables and change the “DB2 Table Qualifier” from SYSIBM to the qualifier of your copy, as shown in Figure 2-17.

```
************** CICS Interdependency Analyzer for z/OS - V5R3M0 **************
Command ===>

                                                                    More:   - +
DB2 variables for CINT:

COLLECTING DB2 INFORMATION . . . YES  (YES or NO)
DB2 VERSION  . . . . . . . . . . V10
DB2 COMPATIBILITY MODE . . . . . NO   (YES or NO)
DB2 LOAD DATA SET NAME . . . . . SYS2.DB2.V10.SDSNLOAD
DB2 RUNLIB DATA SET NAME . . . . DSNV102F.RUNLIB.LOAD
DB2 SUB SYSTEM . . . . . . . . . DI2F
DB2 TABLE QUALIFIER  . . . . . . IAQUAL
DB2 TABLE OWNER  . . . . . . . . CICSIAD
DB2 BUFFERPOOL FOR INDEX . . . . BP1
DB2 PLAN NAME FOR DSNTEP2  . . . DSNTEP10
DB2 PLAN NAME FOR CICS . . . . . CIUCIC53
DB2 COLLECTION ID  . . . . . . . IACOLL53
DB2 AUTHORIZATION ID . . . . . . PUBLIC
```

Figure 2-17 DB2 variables for CINT

When the configuration is complete, you need to rebind the CINB programs by running the following command:

```
<hlq>.SCIUSAMP(CIUDBNT)
```

2.3 Using the collection ID to manage your data

In this section, we look at these topics:

- What collection ID value should you use
- How to set the collection ID

and how we can use the collection ID to:

- Manage our analysis in the plug-in
- Compare data collected for different collection IDs
- Delete data by collection ID

2.3.1 What value should you assign to my collection ID?

A collection ID is assigned to the CICS IA Dependency data when it is loaded from the VSAM file in to the CICS IA DB2 database. A collection ID can be a string up to 16 characters. It can include numerics and special characters.

The collection ID can be used in a number of different ways and it can depend on:

- Why are you collecting the data
- Your environment

If you are collecting data for a specific project, for example:

- Threadsafe analysis
- A major application upgrade
- CICS upgrade

Then, we suggest that all the data you collect before the change is assigned to the same collection ID during the load job. This setting enables you to analyze the “before” image of the data. When you have performed the changes, recollect the data and assign a new collection ID. This setting enables you to analyze the “after” image of the data and by using the IA plug-in you can compare the “before” and “after” images.

If you are collecting data in a stable production environment, you can use the same collection ID. This setting could be used as your base collection ID. You would only change the collection ID if there is a change to your environment.

If you are collecting data in a volatile development environment where there are many changes, you might want to consider doing a weekly load and using a weekly identification for your collection ID. These settings would allow you to compare weekly and delete data associated with collection IDs that are, for example, 5 weeks old.

CICS IA allows you to load data from your development, test, and production regions into the same IA DB2 database. You can use different collection IDs to distinguish between the data. Again, you can then analyze, compare, and delete by using these collection IDs.

You can also use the collection ID to assist you in analyzing a specific problem by collecting data just in the CICS region and a transaction causing the problem. Then, when you load the data in to the IA DB2 database, use a collection ID that relates to that transaction and problem.

2.3.2 How to assign a collection ID

The collection ID for the dependency data is done during the DB2 load. It is used in the following jobs:

- CIUUPDB: Load all dependency tables
- CIUUPDB1: Load CICS and detail dependency tables
- CIUUPDB2: Load DB2 dependency tables
- CIUUPDB3: Load MQ dependency tables
- CIUUPDB4: Load IMS/DLI dependency tables
- CIUUPDBN: Load Natural/ADABAS dependency tables

The setting of the variable depends on your level of JES2. If you are on JES2 z/OS 2.1, we use the new **SYMLIST** parameter feature as shown in Figure 2-18. In this case, you only need to change it in one place.

```
//  EXPORT SYMLIST=(COLL)
//  SET    COLL='_collid_'
....
//STEP000  EXEC PGM=IKJEFT1B,
//            DYNAMNBR=20
//SYSTSIN  DD *,SYMBOLS=EXECSYS
DSN SYSTEM(DI2F)
RUN PROGRAM(CIUUREG) -
    PLAN(IAINC53B) LIB('ANTZ.CICS.IA.DEV.BSF.SCIULOAD') -
    PARMS('DEP,COLLID=&COLL')
END
/*
```

Figure 2-18 Setting the collection ID in JES2 zOS V2.1

If you are on an earlier release of JES2, you need to change the collection ID in several places. This change is best done by using the **CHG ALL** command as shown in Figure 2-19.

```
EDIT       CICSIA51.SCIUSAMP.CICS.TEST(CIUUPDB) - 01.01            Columns 00001 00080
Command ===> chg _collid_ CICS510 all                               Scroll ===> CSR
****** ********************************* Top of Data ********************************
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  129 Line(s) not Displayed
000128 DSN SYSTEM(DI2F)
000129 RUN PROGRAM(CIUUREG) -
000130     PLAN(IAINC51B) LIB('CTSTOOLS.CIA510.SCIULOAD') -
000131     PARMS('DEP,COLLID=_collid_')
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  110 Line(s) not Displayed
000242 DSN SYSTEM(DI2F)
000243 RUN PROGRAM(CIUU055) -
000244     PLAN(IAINC51B) LIB('CTSTOOLS.CIA510.SCIULOAD') -
000245     PARMS('UPDATE,COLLID=_collid_')
000246 END
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   55 Line(s) not Displayed
```

Figure 2-19 Change all collection IDs

2.3.3 Using the collection ID to manage our analysis

All of the collection IDs you assigned to the CICS IA database are displayed in the upper left side of the CICS IA perspective, as shown in Figure 2-20.

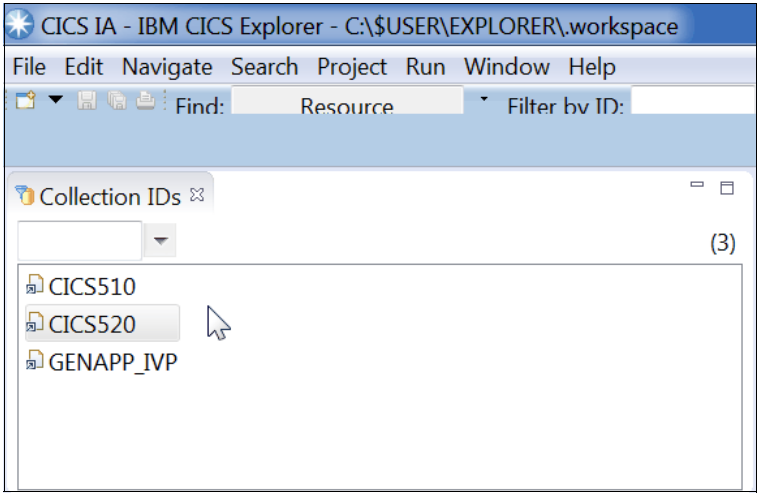


Figure 2-20 The Collection IDs view

In this section and following sections, we use these collection IDs to perform the following functions:

- Manage the amount of data returned
- Compare data across collection IDs
- Delete data associated with a collection ID

To manage the data you analyze and the amount of data returned when using the CICS IA plug-in, you can set a collection ID scope. When you set a specific scope, all the analysis you do including queries and reports will be restricted to the collection ID scope you selected. Setting the scope to a specific collection ID has two advantages:

- Reduces the amount of data returned over your JDBC connection
- Clarifies your analysis

To set a scope, right-click the collection ID you require and select “Set as current scope,” as shown in Figure 2-21.

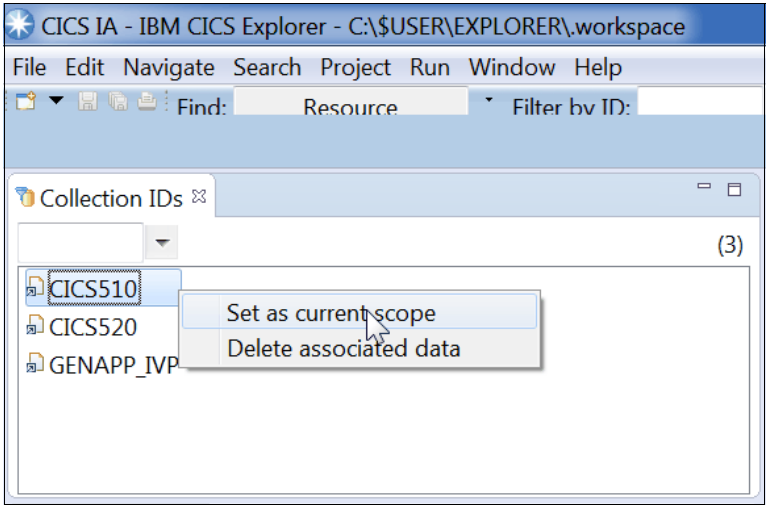


Figure 2-21 Set collection ID CICS510 to be the current scope

The collection ID you selected is displayed in the menu as shown in Figure 2-22.

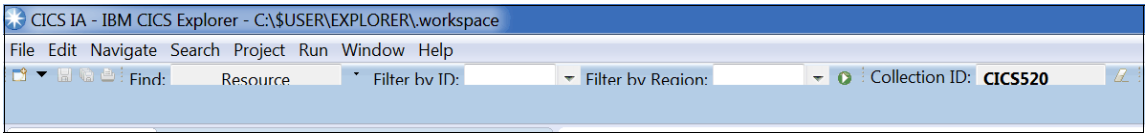


Figure 2-22 Collection ID is set to CICS520

You can clear the collection ID scope by clicking the “erase” icon as shown in Figure 2-23.

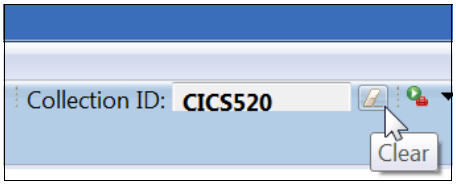


Figure 2-23 Clear the collection ID

As mentioned earlier, by selecting a scope you reduce the amount of data you analyze. You can see this reduction in the data that is shown in the `Regions` folder and the Transactions or Program views.

In Figure 2-24, there is no scope set and you see all of the collected data. You can see four CICS regions and eight transactions.

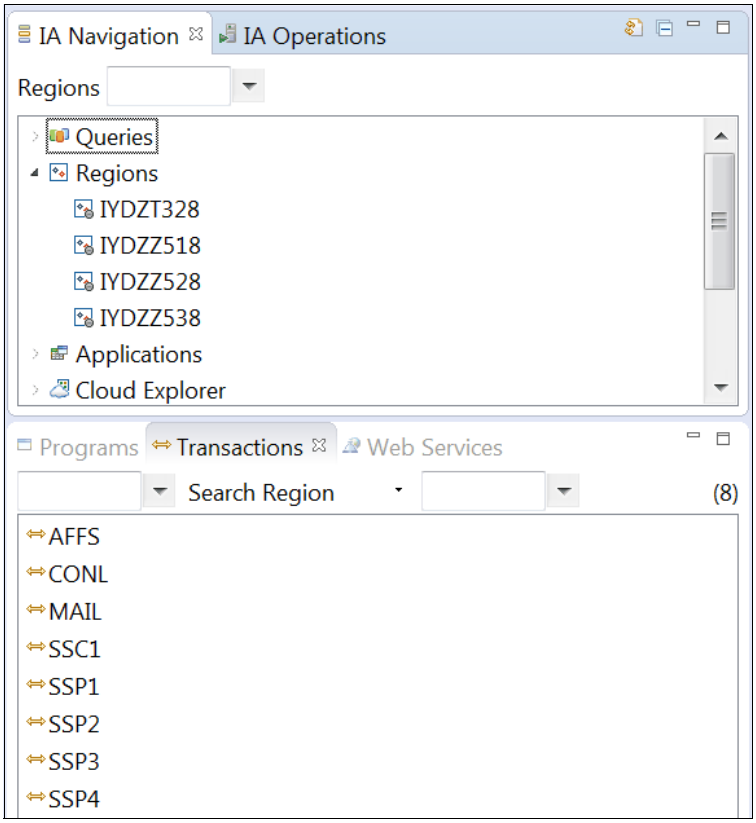


Figure 2-24 No collection ID scope

If you now set the scope to be the IVP collection ID, the amount of data returned in these views is reduced as shown in Figure 2-25.

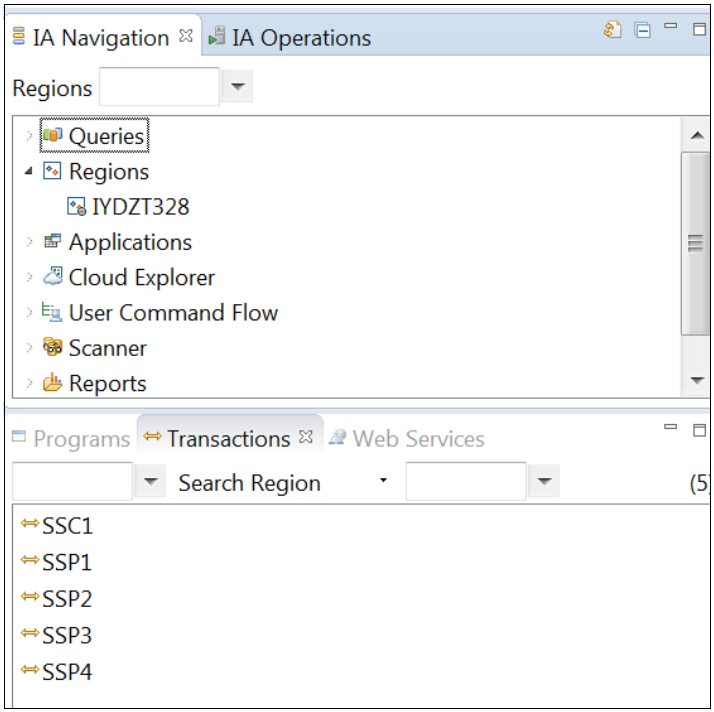


Figure 2-25 Collection ID scope set to GENAPP_IVP

The scope is also applied when analyzing data, running reports, and running queries, for example, if we select to see the resources used by transaction SSC1 and select a specific region as shown in Figure 2-26.

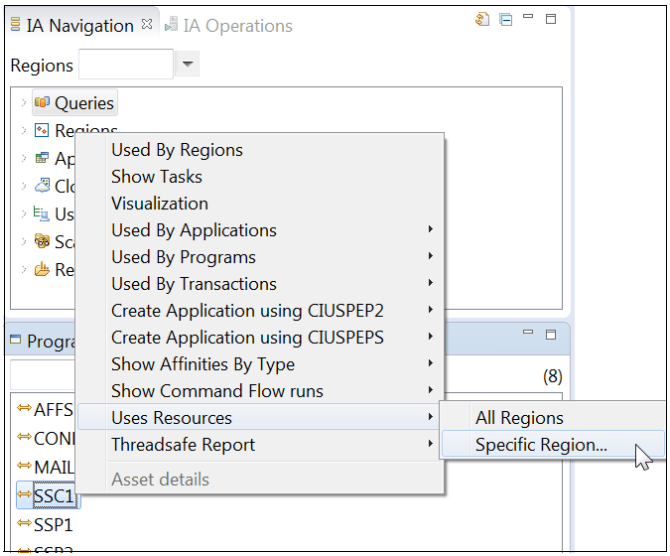


Figure 2-26 Show resources used by SSC1 for a specific region

If there is no collection ID scope set, all of the CICS regions where data for transaction SSC1 will be selectable as shown in Figure 2-27.

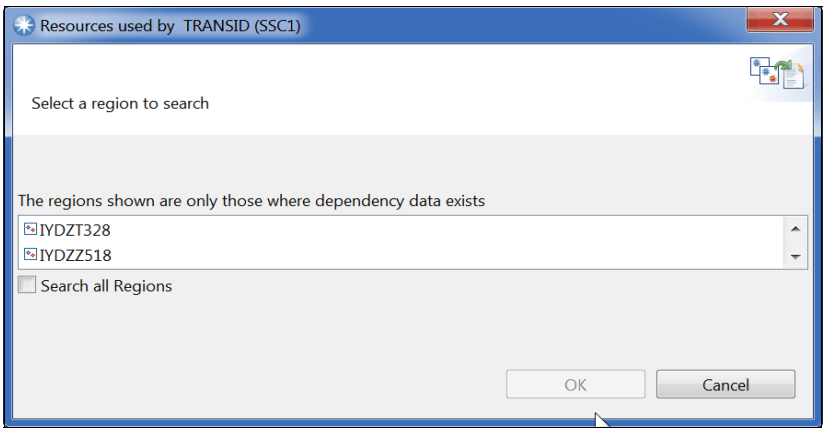


Figure 2-27 No collection ID scope set

If the collection ID scope is set to GENAPP_IVP, only one CICS region is selectable as shown in Figure 2-28.

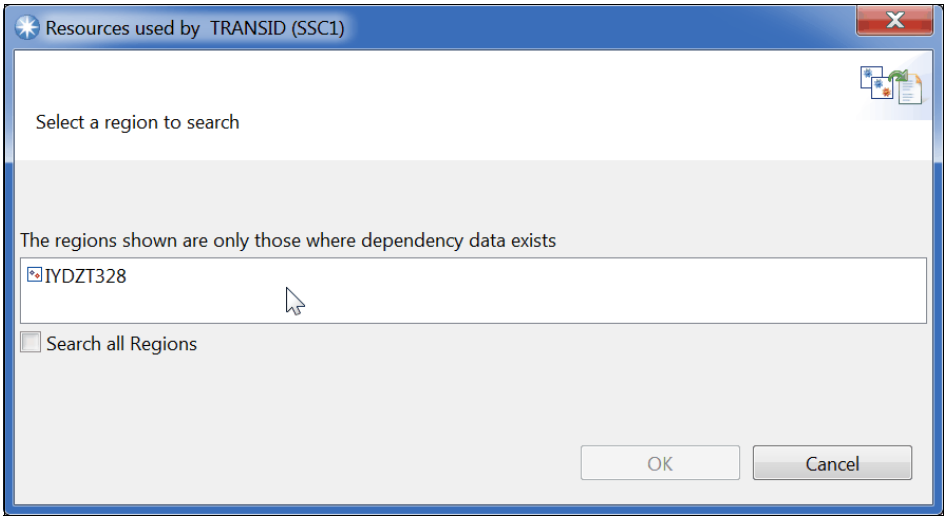


Figure 2-28 Collection ID scope set to GENAPP_IVP

If you run a threadsafe report with the collection ID set to GENAPP_IVP, the scope is associated with the report as shown in Figure 2-29.

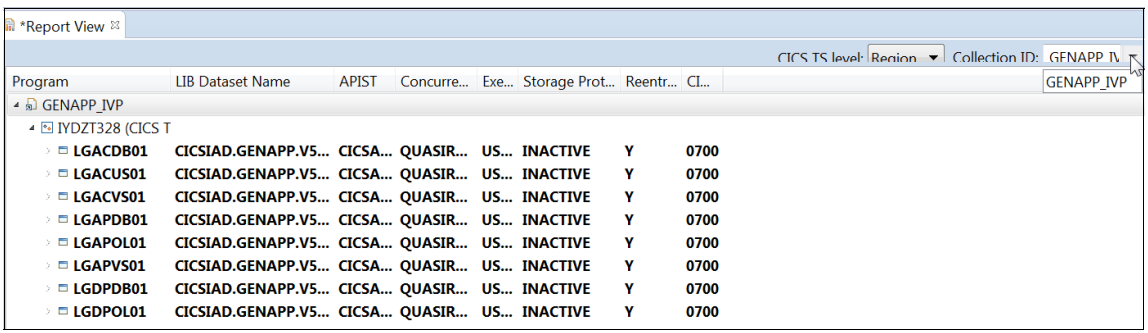


Figure 2-29 Threadsafe report for collection ID GENAPP_IVP

2.3.4 Comparing collection IDs

In this section, we run a query against different collection IDs, save the results, and then compare them.

We use a simple query to list the programs used as shown in Figure 2-30. You could use this type of query to determine the amount of testing you have done when you are upgrading your CICS system. In this example, we collected automated test data in our CICS V510 region and set the collection ID to CICS510. We have all also collected the data in our CICS V520 region and set the collection ID to CICS520. By comparing the output, you can see which programs you have not tested against V520.

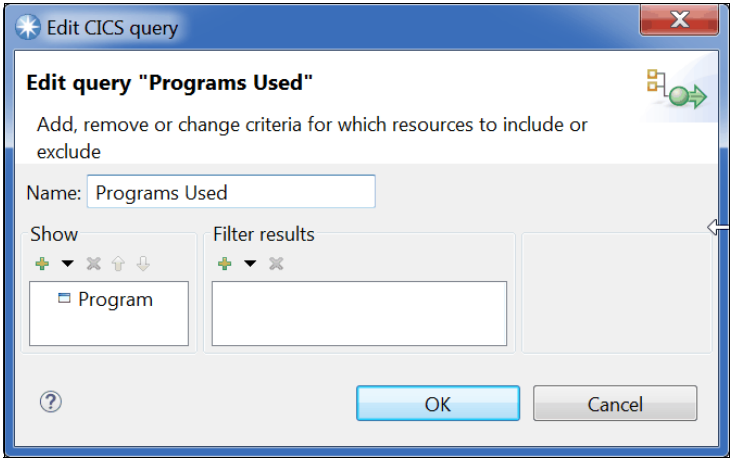


Figure 2-30 Simple query to list programs used

We now set the scope to CICS510 and run the query. The output is shown in Figure 2-31. There are 28 programs (shown in the upper-right corner) and this query has a scope of CICS510 (shown in upper-left corner).

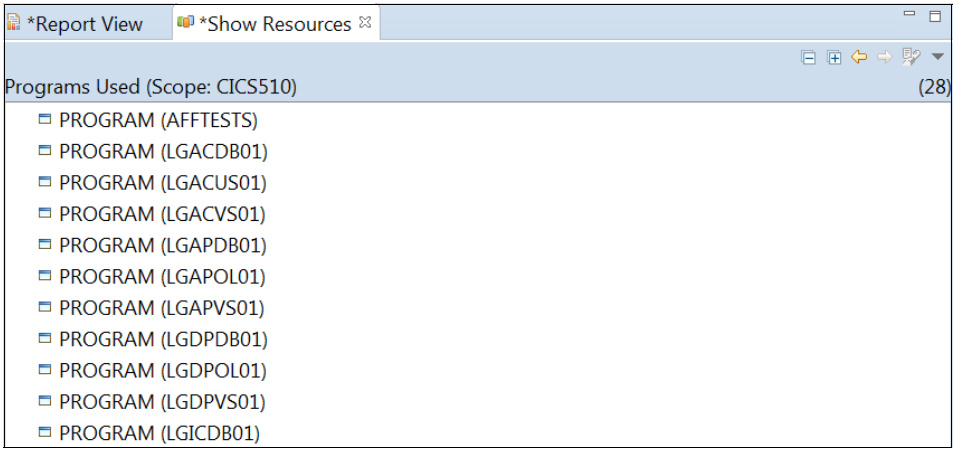


Figure 2-31 Programs used in CICS510

We now save the output. Select **File → Save** as shown in Figure 2-32 and enter a meaningful name for your query as shown in Figure 2-33.

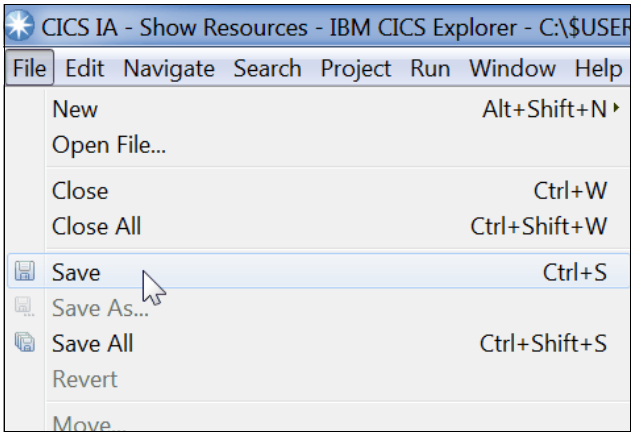


Figure 2-32 Save the results of your query

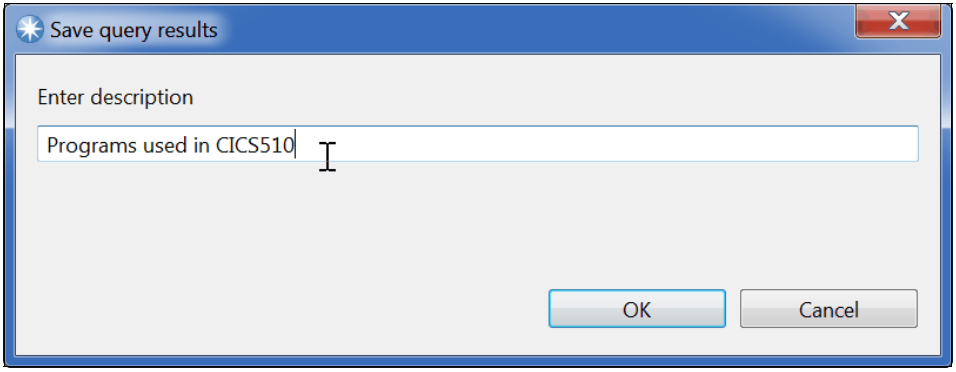


Figure 2-33 Enter a query description

We now switch the collection ID scope to CICS520 and rerun the query and the save.

In the `Queries` folder, we can now see that we have two results saved for the Programs Used query as shown in Figure 2-34.

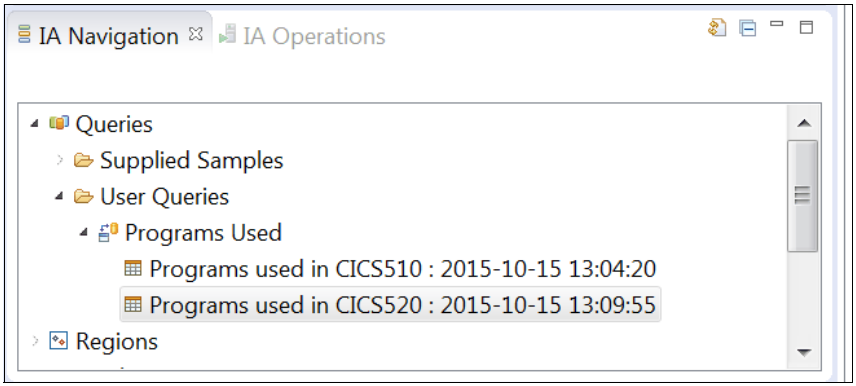


Figure 2-34 Results saved for query "Programs Used"

Note: The date is also stored when the query output is saved. You can use this feature to save the results on a weekly basis as you proceed with your CICS TS upgrade testing.

We can now compare the result sets to see how many programs we have not tested in V520. You need to open both result sets in the Show Resources view and then select the "compare" icon found in the upper-right corner of the view as shown in Figure 2-35.

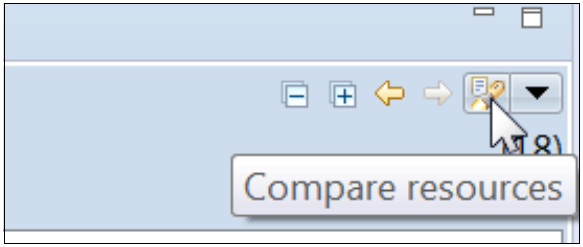


Figure 2-35 Compare results

In "Compare search results," select the results you want to compare as shown in Figure 2-36 on page 55.

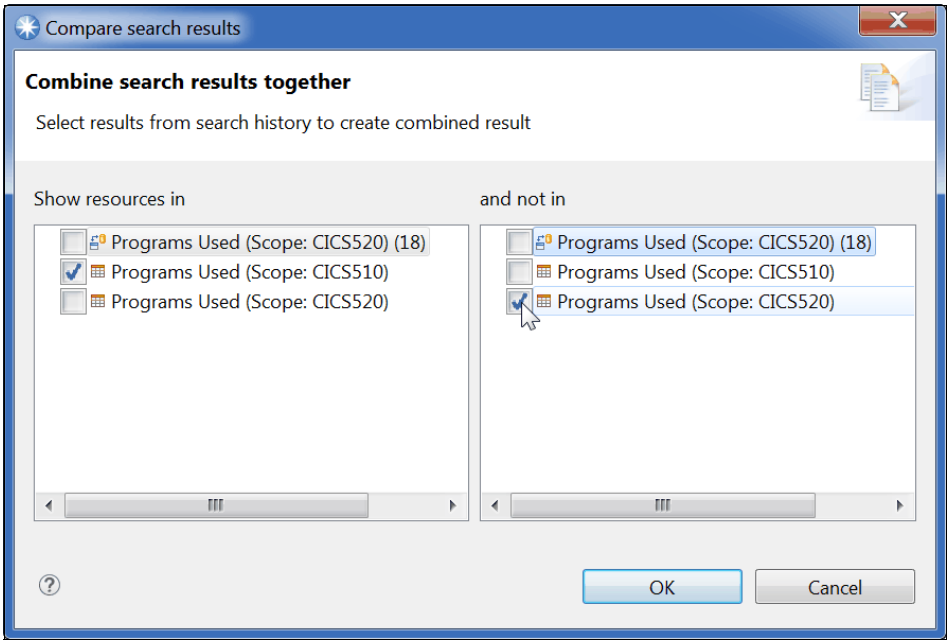


Figure 2-36 Compare programs used in CICS510 and not in CICS520

The output is also displayed in the Show Resources view as shown in Figure 2-37 on page 56. We can see that there are 11 programs used in CICS510 that we have not collected data for in CICS520.

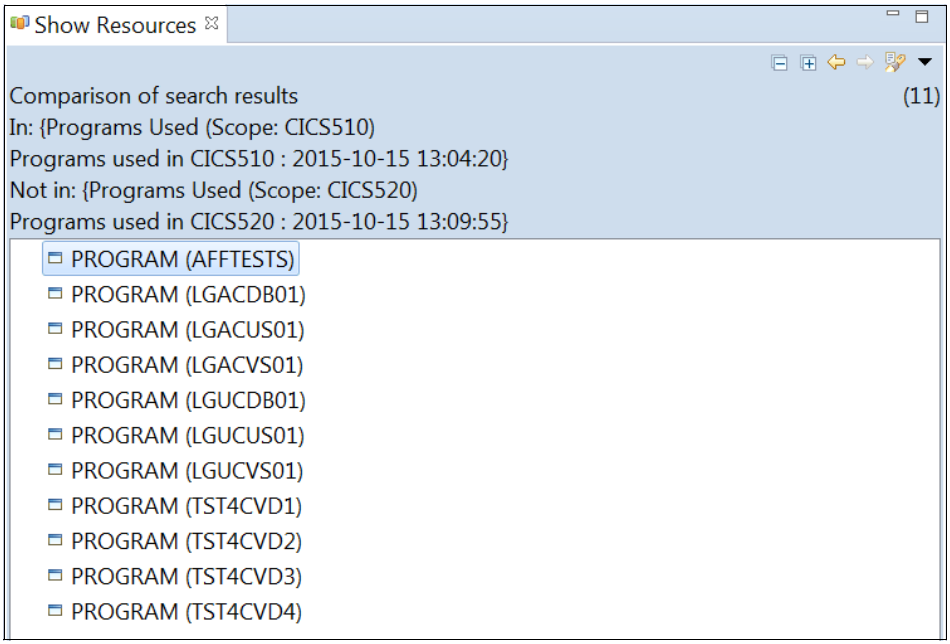


Figure 2-37 Output for compare programs used

2.3.5 Deleting data by collection ID

In this section, we use the CICS IA plug-in to delete the data associated with the IVP collection ID.

Right-click the “GENAPP_IVP” collection ID and select “Delete associated data” as shown in Figure 2-38.

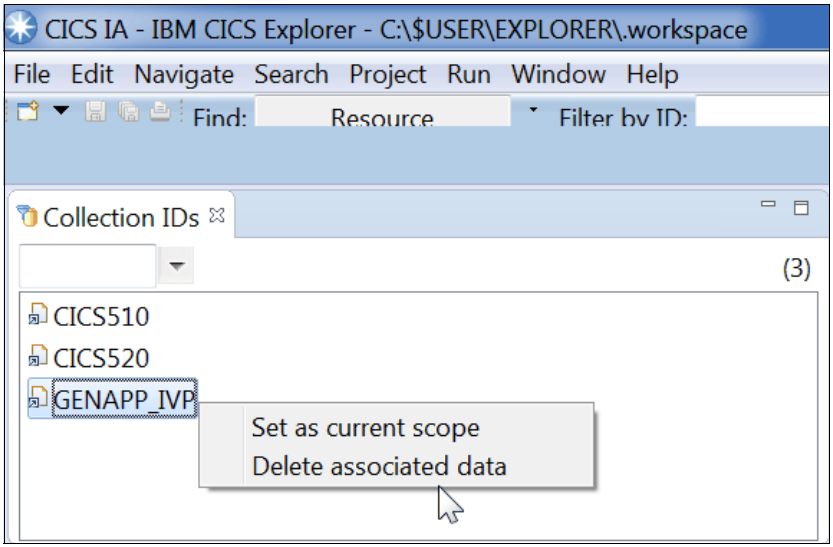


Figure 2-38 Delete data associated with GENAPP_IVP

You get an opportunity to confirm or cancel as shown in Figure 2-39.

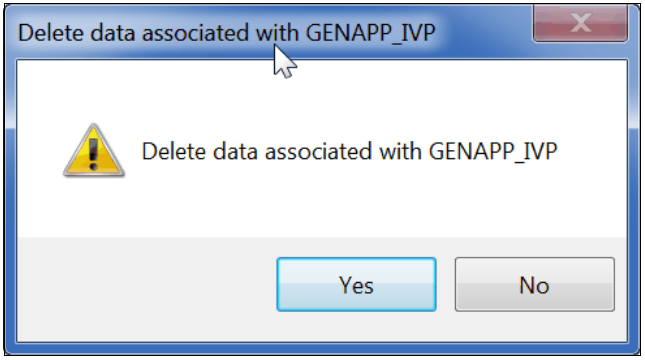


Figure 2-39 Confirm the delete

Select **Yes** to confirm the delete and the data is deleted and a deletion report is given as shown in Figure 2-40 on page 58.

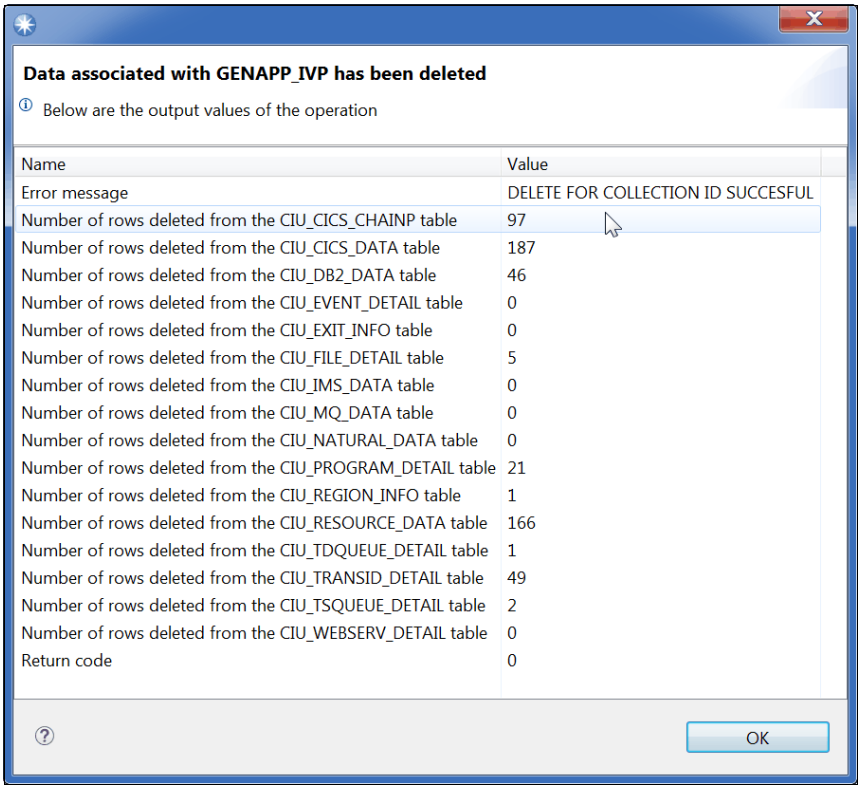


Figure 2-40 Data associated with GENAPP_IVP has been deleted

3

Using CICS IA daily

In this chapter, we look at how CICS Interdependency Analyzer (IA) can be used on a daily basis to understand the impact analysis of code changes.

3.1 Day-to-day usage

In this section, we look at how CICS IA can be used daily to assist with analyzing the impact of application changes. We also cover how CICS IA can be used to assist with the following actions:

- Governance: Reviewing code changes while deploying from development to test to production
- Operations: Understanding operation resource relations
- Abend analysis: Using IA data to understand changes

3.1.1 Impact analysis of application changes

In this section, we see how the CICS IA Explorer plug-in can quickly give you an understanding of the CICS regions, transactions, and programs that are impacted when a developer changes a program. A better understanding of this impact can lead to quicker changes with less risk.

We demonstrate how CICS IA can help by using example use cases.

Use Case 1

As an application developer, I have been asked to add a field to my customer file, KSDSCUST. Before I start what do I need to know?

What programs use file KSDSCUST?

Use Case 2

As a tester, I have been asked to test the changes to the customer file, KSDSCUST. Before I start what do I need to know?

- Which transactions do I need to run to test file KSDSCUST?
- Which CICS regions can I use to test my changes?

Use Case 3

As a systems administrator, I have been asked to deploy the changes to the customer file, KSDSCUST, into production. Before I start what do I need to know?

- Which other resources do the changed transaction impact?
- Have any “site” standards been broken?

Let us look at how CICS IA can help with these questions.

First, we need to know if we collected data for file KSDSCUST. In the CICS IA plug-in, we can display all resources collected from the search bar as shown in Figure 3-1.

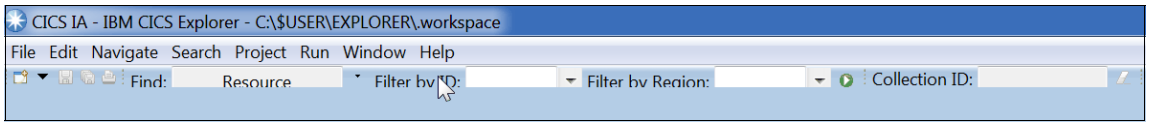


Figure 3-1 The CICS IA resource find bar

You can filter this search by resource type, by the resource name, and the CICS region. In this case, we find all FILE resources starting with KSDS as shown in Figure 3-2.

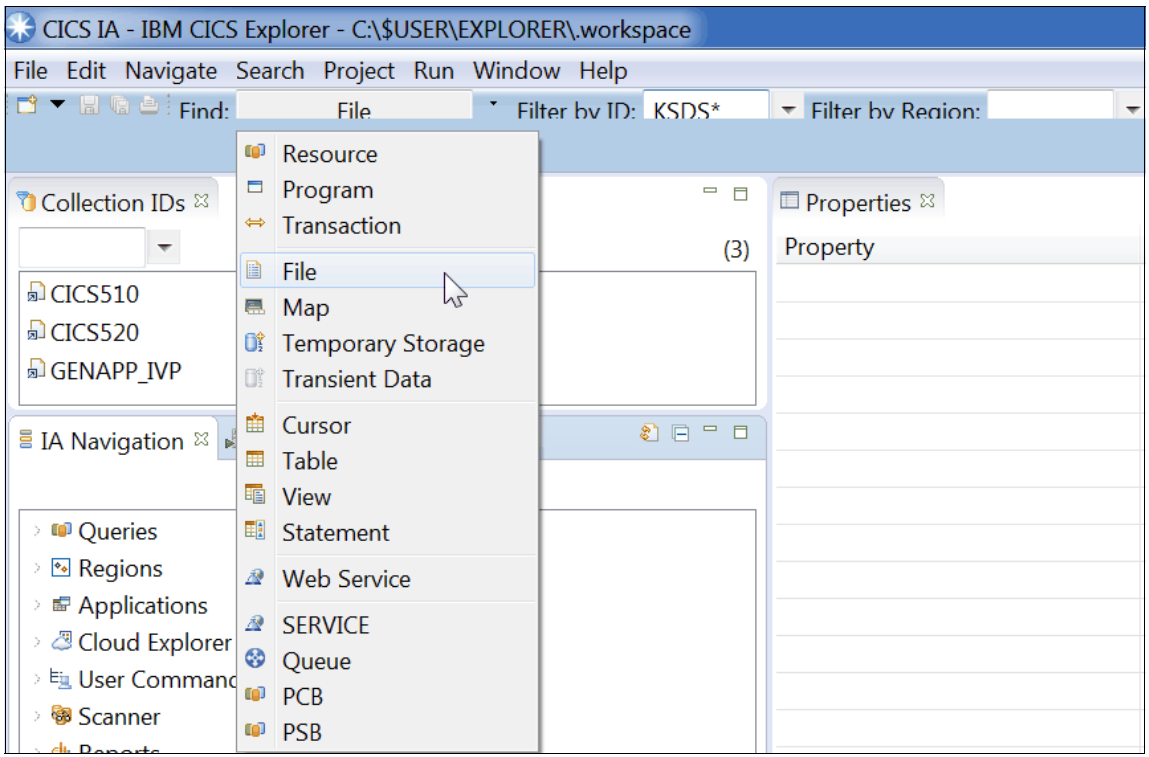


*Figure 3-2 Find all files starting with KSDS**

When you set your filters, select the green run arrow. The results are displayed in the Show Resources view as shown in Figure 3-3.

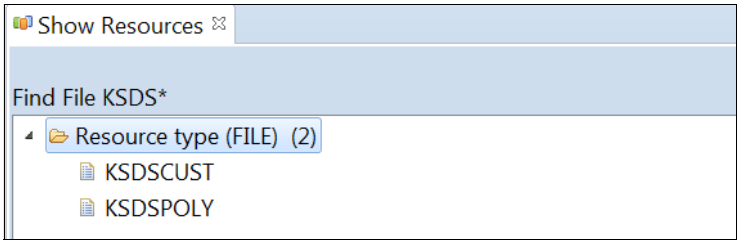


Figure 3-3 Results for a search for all files starting with KSDS

We can see that there are two files: KSDSCUST and KSDSPOLY. We are interested in the KSDSCUST file. We can now start to answer some of our questions.

What programs use file KSDSCUST?

To find out which programs use this file, right-click KSDSCUST and select **Used by Programs → All Regions** as shown in Figure 3-4.

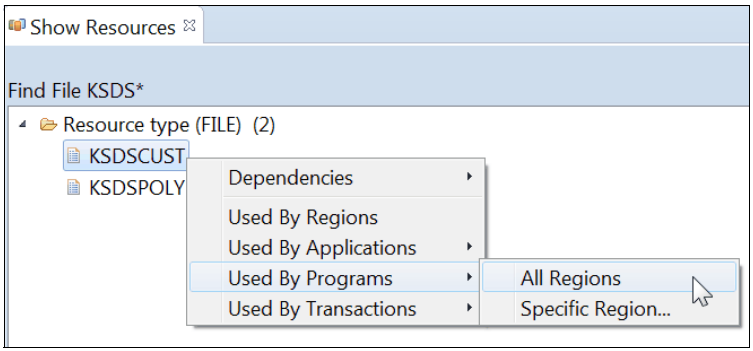


Figure 3-4 Which programs use file KSDSCUST?

The results of this question are shown in the Used By view as shown in Figure 3-5 on page 63.

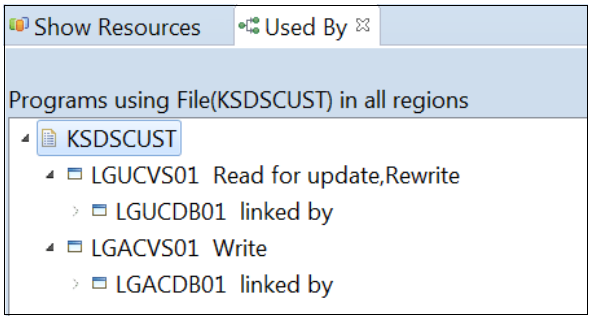


Figure 3-5 Programs using file KSDSCUST

We can quickly see that program LGUCVS01 issues a “Read for update” and a “Rewrite” against this file; program LGACVS01 issues a “WRITE” against the file.

We can also see some information about how these programs were invoked. For example, program LGUCDB01 issued an EXEC CICS LINK to program LGUCVS01.

What transactions use file KSDSCUST?

As we did before to find out which transactions use this file, right-click KSDSCUST and select **Used by Transactions → Specific Region**. In this case, we select our region as shown in Figure 3-6.

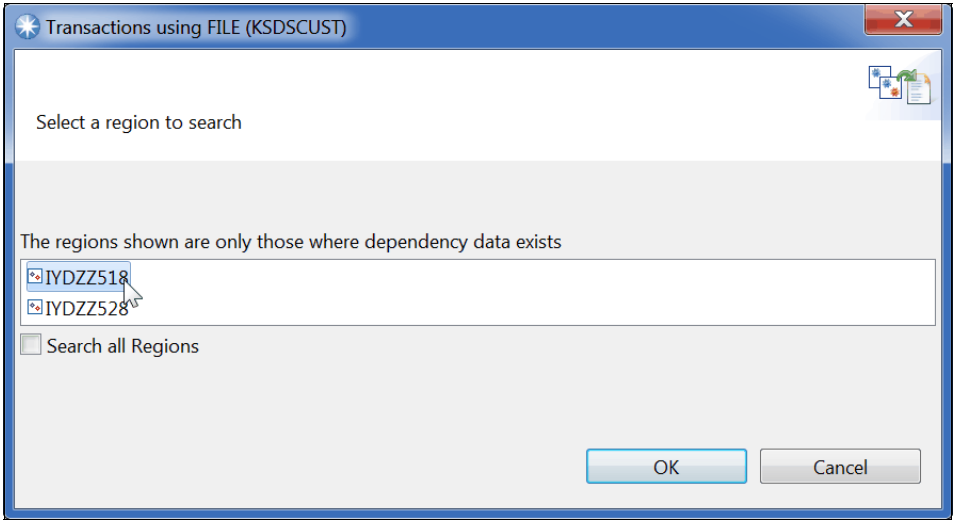


Figure 3-6 Choose your CICS region

Again, the results are shown in the Used By view, as shown in Figure 3-7.

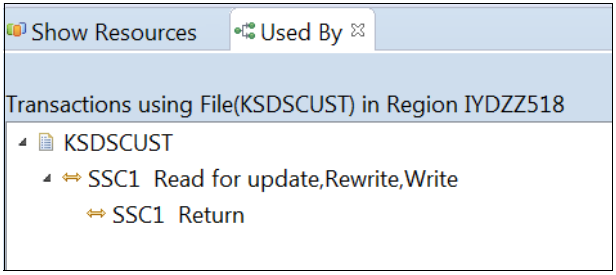


Figure 3-7 Transactions using file KSDSCUST

Note: Your previous Used By view for your program query is not lost. It can be restored by using the left and right yellow arrows in the upper-right corner of the view as shown in Figure 3-8.

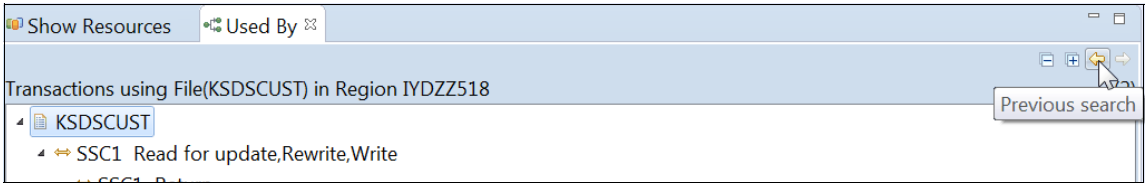


Figure 3-8 Left and right view arrows

These arrows can be found in many of the CICS IA views including the Show Resources view.

In what regions can I test the changes to file KSDSCUST?

We already answered this question by asking for which transactions use the file in a specific region, as shown in Figure 3-6 on page 63. You can also ask a specific query by right-clicking the file and selecting “Used by Regions” as shown in Figure 3-9 on page 65.

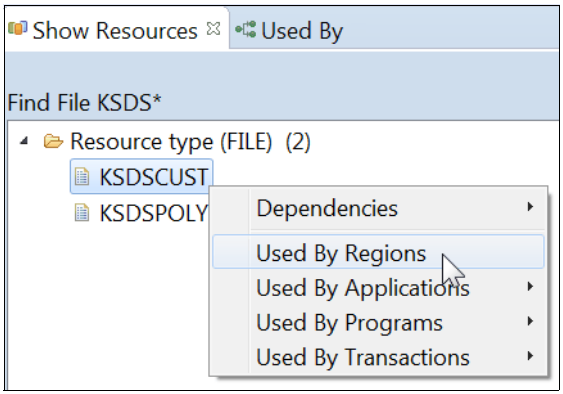


Figure 3-9 Used by Regions selection

In this case, the results are shown in the Show Resources view as shown in Figure 3-10.

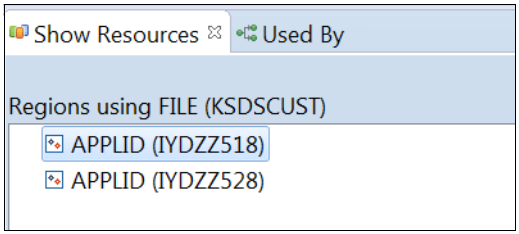


Figure 3-10 CICS regions using FILE KSDSCUST

Note: You can use the right and left yellow arrows to scroll through and see the results for all your questions.

Which other resources does the changed transaction impact?

We already discovered that the KSDSCUST file is used by transaction SSC1, but before deploying this change it would be good to understand the other resources used by this transaction.

From the Used By view, right-click the SSC1 transaction and select **Uses Resources → Specific Region** as shown in Figure 3-11.

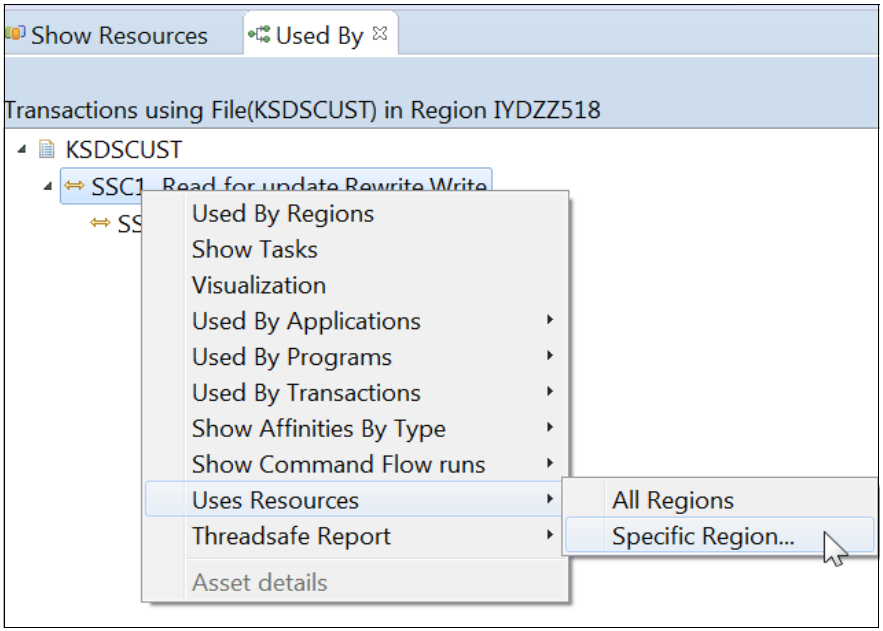


Figure 3-11 Which resources do transaction SSC1 use

You can then select your region and the results are shown in another new view, the Uses view, as shown in Figure 3-12 on page 67.

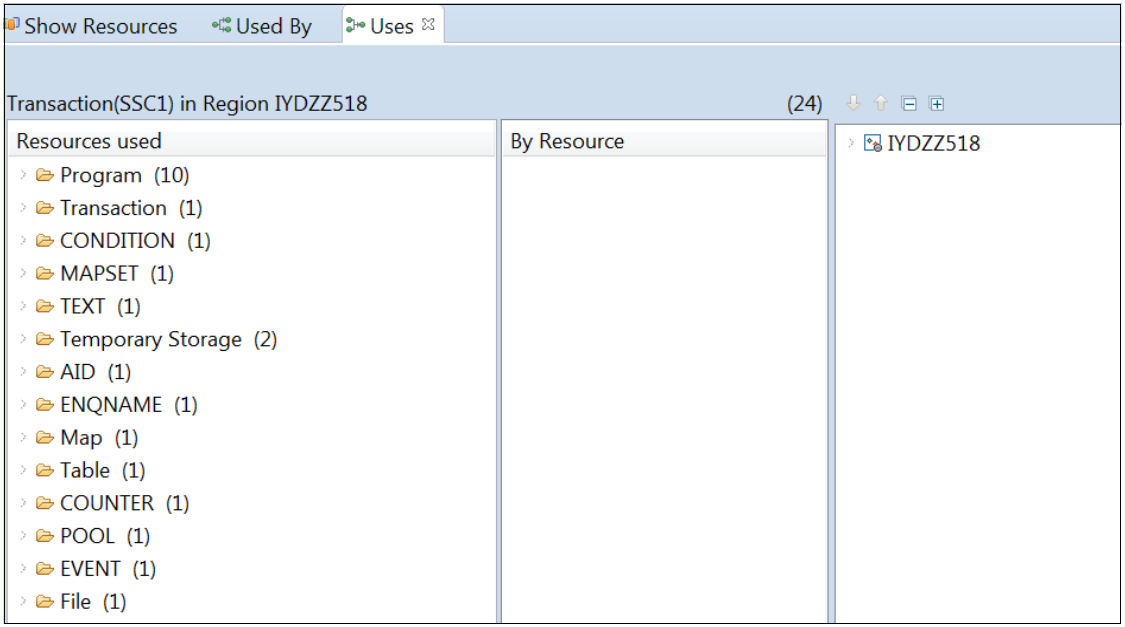


Figure 3-12 The Uses view for transaction SSC1

3.1.2 The Uses view

The Uses view is split into three different areas:

- Resources used
- By program
- Program flow

The “Resources used” view lists the resources used by this transaction. The resources are grouped by resource type and can be expanded to show the resource name and the commands issued against that command as shown in Figure 3-13.

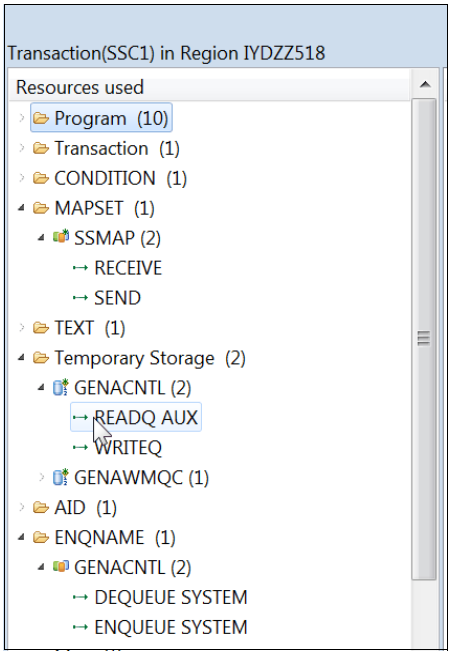


Figure 3-13 Resources used by SSC1

You can use the toggle icon in the upper-right corner to show the resources by command followed by resource name as shown in Figure 3-14 on page 69.

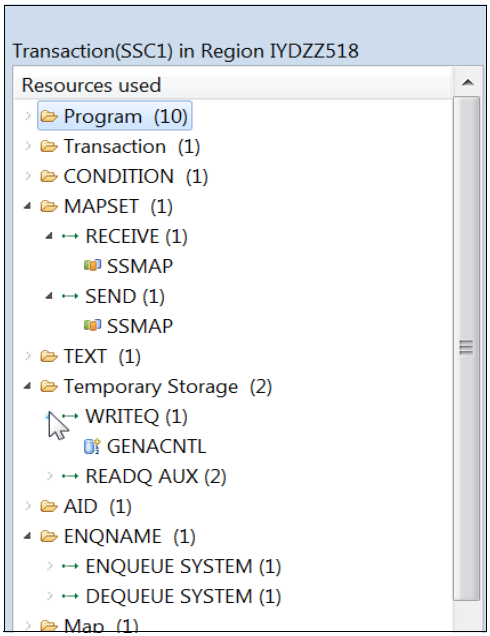


Figure 3-14 Resources used by SSC1: Toggled to show commands

The “By resource” section lists the specific programs that use a resource or a command when selected in the “Resources used” section. For example, you can quickly see which programs use TSQUEUE GENACNTL by selecting this resource and looking in the By Program section as shown in Figure 3-15.

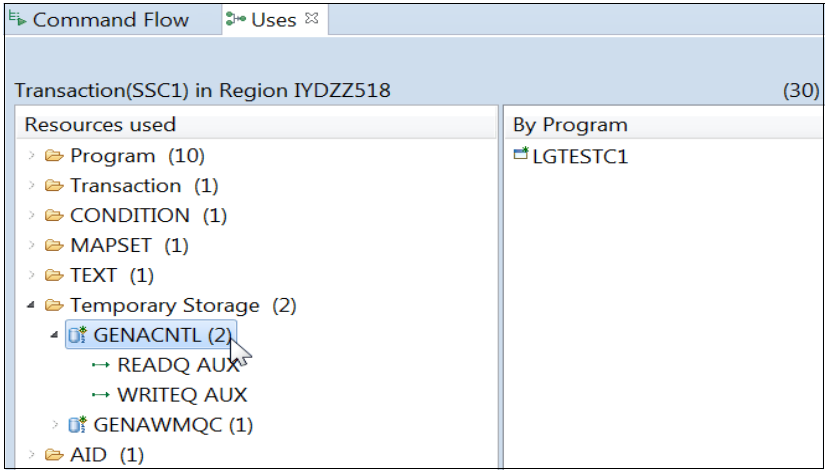


Figure 3-15 TSQUEUE GENACNTL is used by program LGTESTC1

The third and rightmost section of the Uses view displays the program flow.

Note: The flow is *not* in chronological order.

It shows CALLs, LINKs, and XCTLs issued by programs. Again, if you select a resource in the Resources used section, the programs that use it are shown in both the By Program section and the flow section as shown in Figure 3-16. This example shows that the file KSDSCUST is used by two programs, LGUCVS01 and LGACVS01, as shown in the By Program section. The flow section highlights where program LGUCVS01 is within the overall flow of the SSC1 transaction.

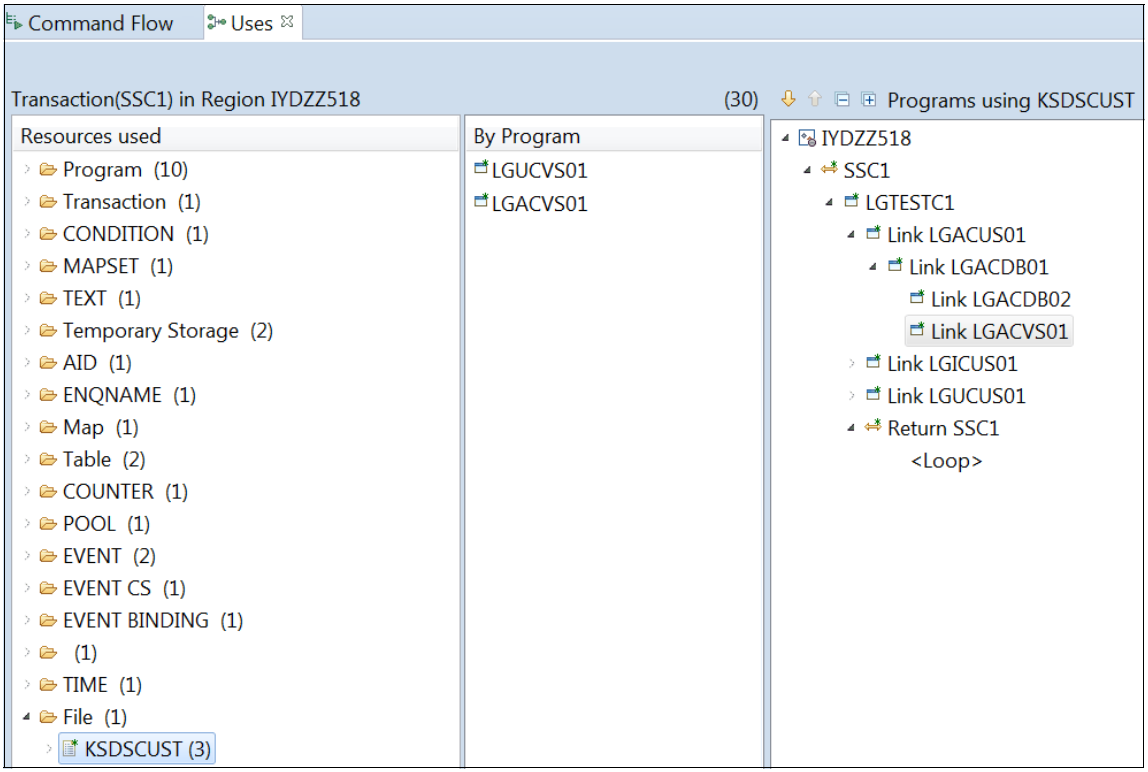


Figure 3-16 Program flow for transaction SSC1: Highlighting KSDSCUST file usage

3.1.3 Using a user query to govern site standards

As a systems administrator, I want to make sure that the latest development work has not introduced any commands that are not allowed to be deployed into production. For example, many COBOL developers use EXEC CICS ASKTIME and the COBOL DISPLAY command for diagnostic purposes during testing and usually forget to remove them. You can use CICS IA to check for these commands. There are a number of methods to perform this check.

We look at two methods.

Method 1

You know that changes have been made to transaction SSC1 so you can simply show resources used by this transaction and look for the **DISPLAY** and **ASKTIME** commands as shown in Figure 3-17.

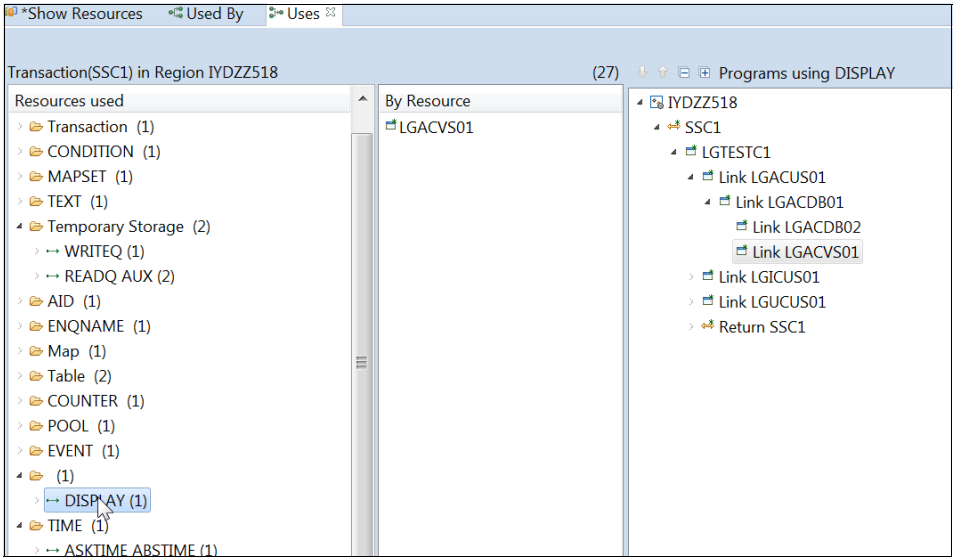


Figure 3-17 A DISPLAY command is issued by program LGACVS01

Method 2

In this method, we introduce the CICS IA Query Editor feature to create a generic query to look for **DISPLAY** and **TIME** commands. The query editor allows you to generate simple SQL to answer more specific questions than the IA plug-in resource views allow.

The IA plug-in is shipped with “supplied queries,” which are ready for immediate use. These queries can assist with CICS TS upgrades and other common questions that are relevant to most customers. These queries can be found in the Navigation view under the `Queries` folder as shown in Figure 3-18.

This feature also allows you to create your own user queries, which can be more specific to your configuration and to your needs. We create a user query to display programs that issue the **`DISPLAY`** command.

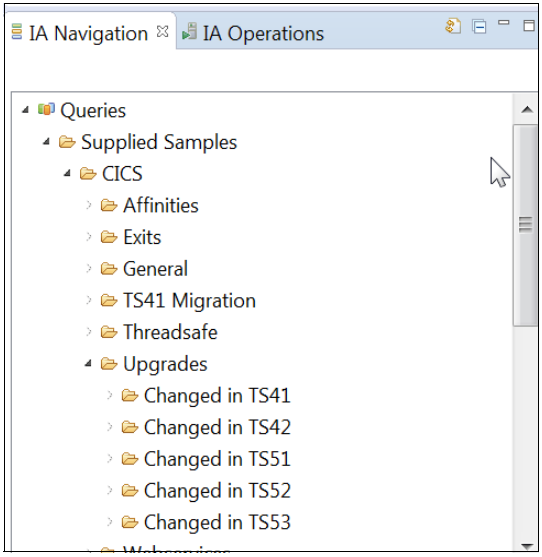


Figure 3-18 IA supplied queries

In the Navigation view, right-click the `User Queries` folder and select **New → CICS query** as shown in Figure 3-19 on page 73.

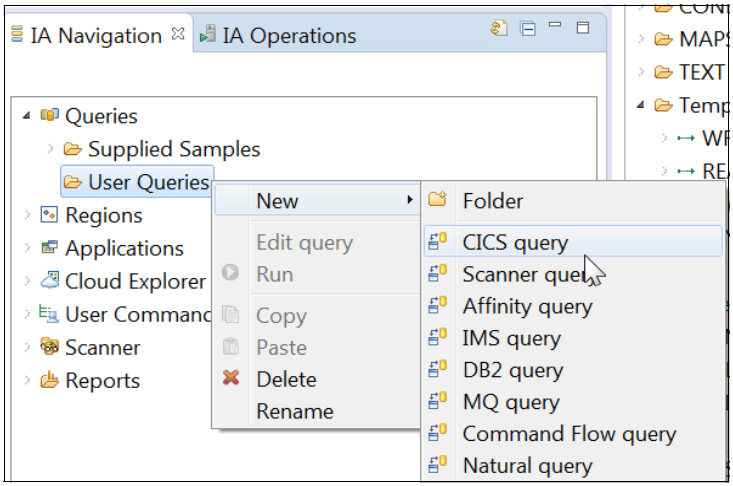


Figure 3-19 Create a CICS query

The Query window editor opens as shown in Figure 3-20.

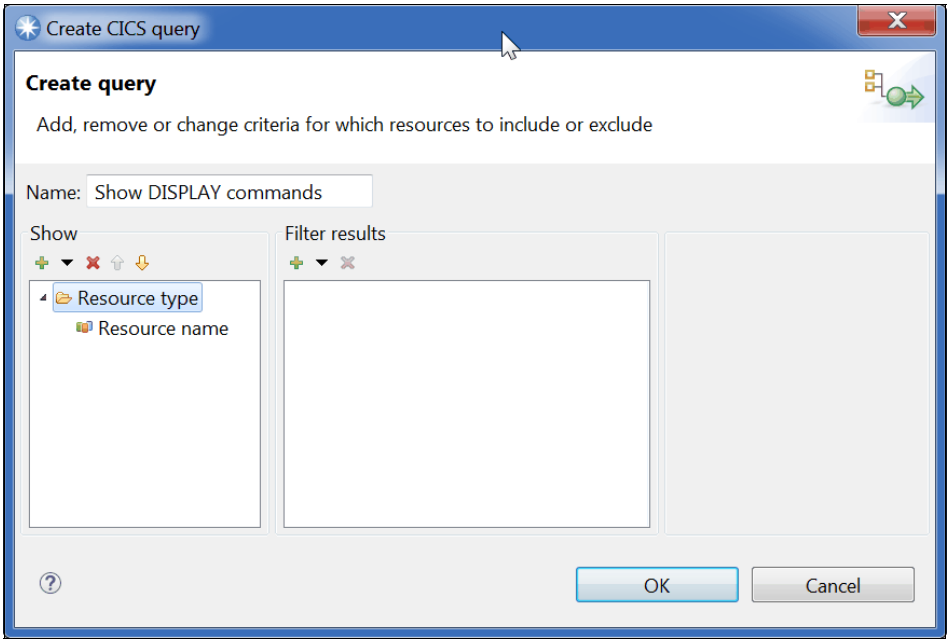


Figure 3-20 Query editor window

The editor is broken into four input areas:

- Name
- Show
- Filter results 1
- Filter results 2

The Name input field allows you to name your query. For example, “Show DISPLAY command”.

The Show input area allows you to select which columns to display in the output. You can use the pull-down arrow to select the columns you want as shown in Figure 3-21.

We select the Applid, Transaction, Program, and Command columns. You can use the yellow up and down arrows to put the columns in the order you want. You can use the red “X” icon to remove columns.

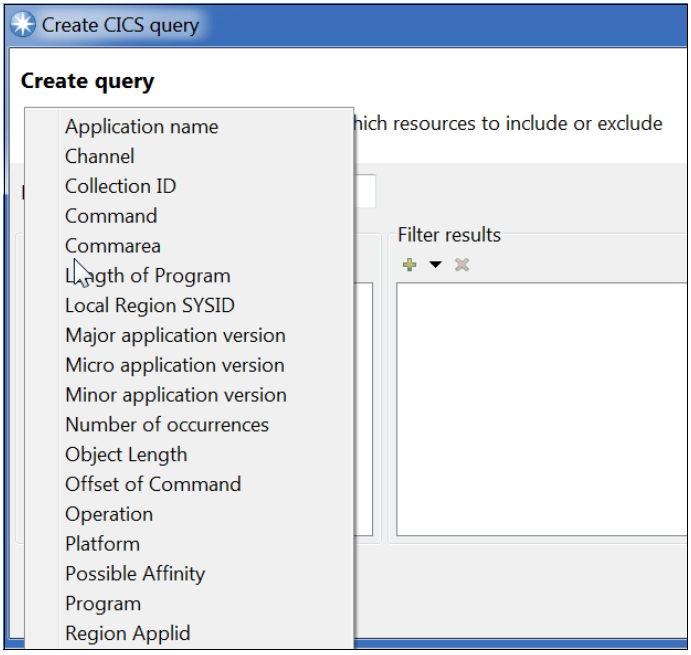


Figure 3-21 Choose the columns to display

The “Filter results” input area allows you to set your column filters. This option corresponds to the WHERE clause in an SQL SELECT. Again, you can use the pull-down arrow, the red “X” icon, and the yellow up and down arrows. In this case, we filter by the COMMAND column where the function is a DISPLAY command. First, we use the pull-down arrow to select the COMMAND column.

This action then adds in the filter results input area, which allows you to set the filter. This area is specific to the column you select. In this case, we selected the COMMAND column and the input area allows us to select which commands we want to include, as shown in Figure 3-22.

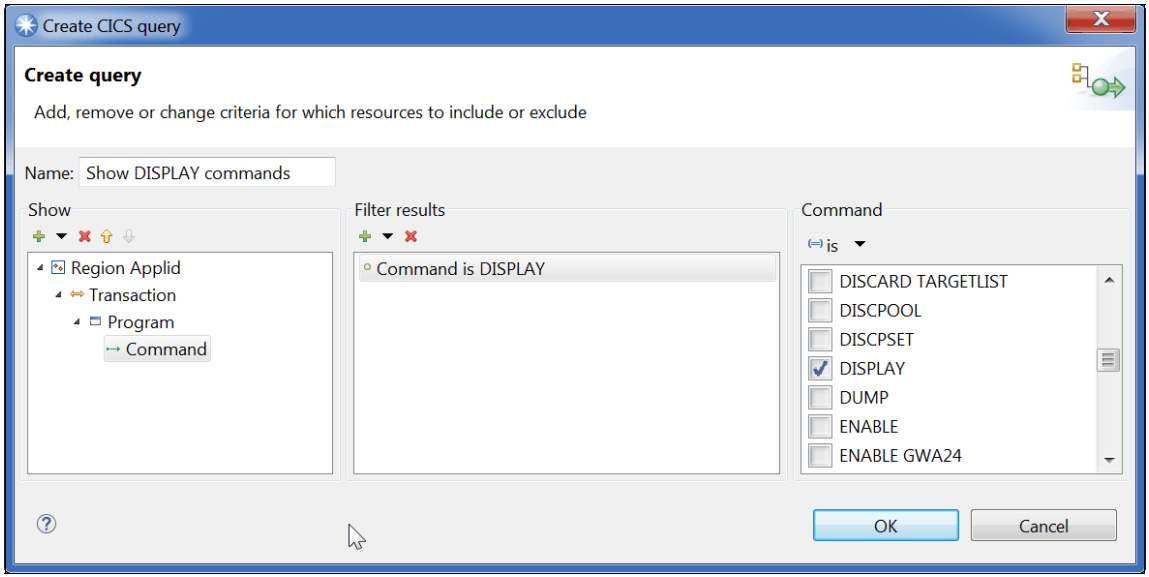


Figure 3-22 User query to list programs that issue a DISPLAY command by region and transaction

Click **OK** to save this command in the `User Queries` folder. You can then run the query by double-clicking or right-clicking and selecting “Run” as shown in Figure 3-23.

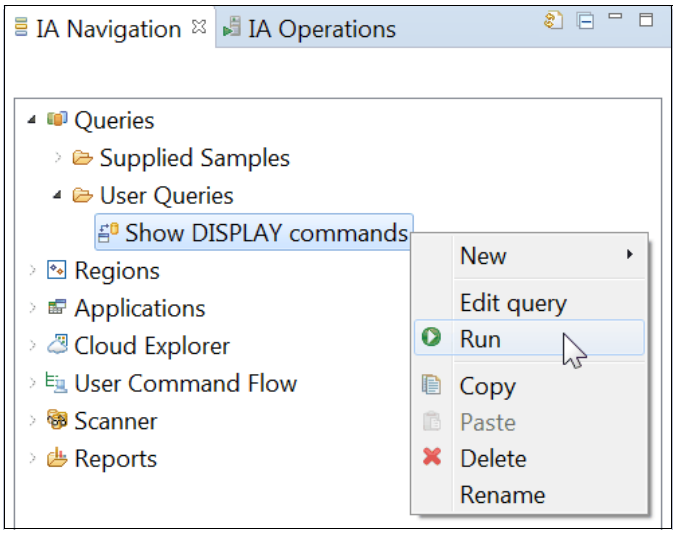


Figure 3-23 Running a query

The results are shown in the Show Resources view as shown in Figure 3-24. We can see that the modified program LGACVS01 is using a display command.

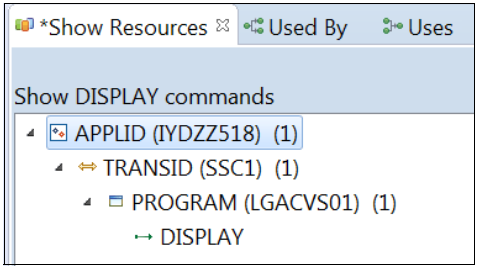


Figure 3-24 Results of the user query

3.1.4 More on IA plug-in user queries

There are several more things that you can do with user queries, including the following actions:

- Saving the results
- Saving the query and pasting the SQL
- Comparing the results

Saving the results

To save the results, select the Show Resources view and then select **File** → **Save** in the upper-left corner as shown in Figure 3-25, or enter CTRL+S.

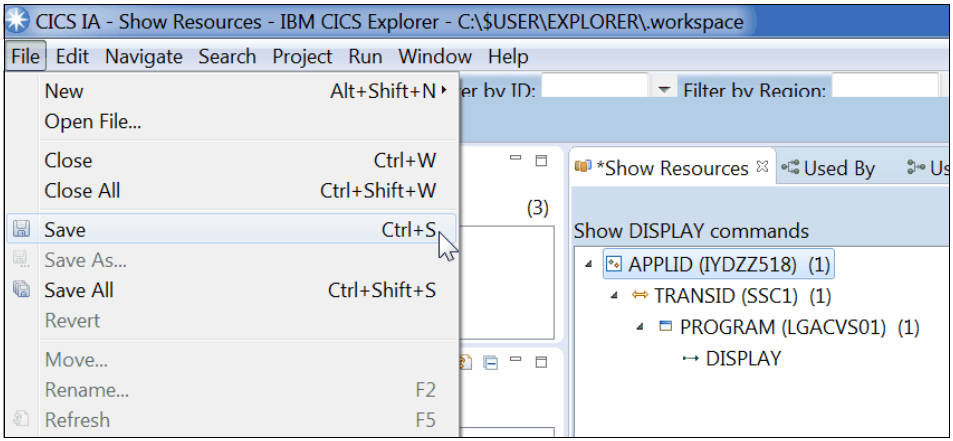


Figure 3-25 Saving a query

This action opens a window where you can enter a description of the query being saved as shown in Figure 3-26.

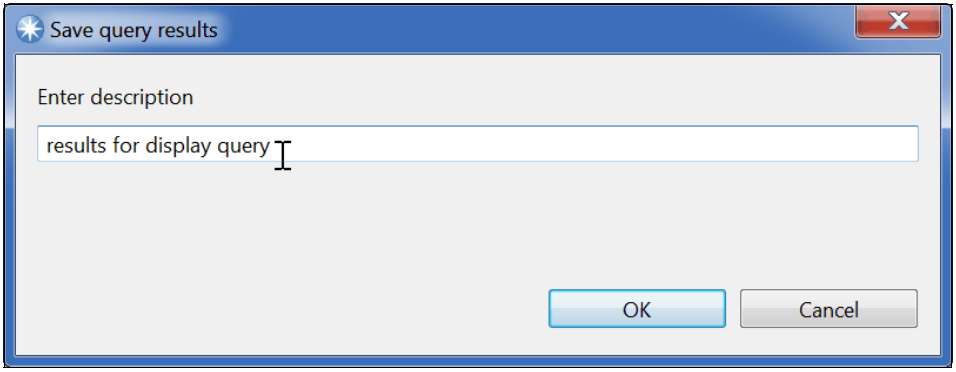


Figure 3-26 Enter a description

Click **OK** and the results are saved under the actual query in the `User Queries` folder as shown in Figure 3-27.

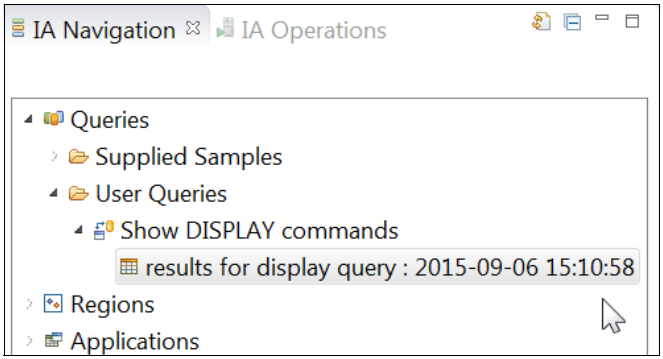


Figure 3-27 Results are saved in the IA Navigation view

The results can then be copied and can be used as input to a spreadsheet product.

Saving the query and pasting the SQL

As well as saving and copying the results, you can copy and paste the SQL that drives the query. Right-click the query and select “Copy” as shown in Figure 3-28 on page 79.

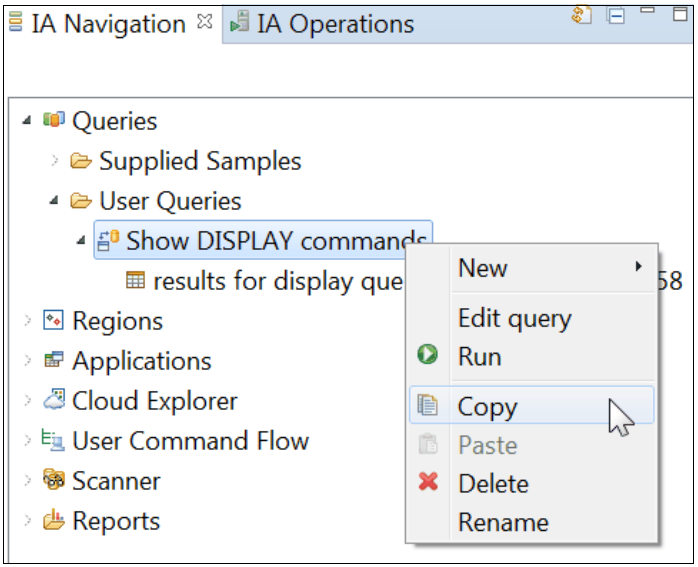


Figure 3-28 Copying the SQL

You can then paste this copy into a document or directly into the DB2 SPUFI tool. In this case, I pasted the query into the book as shown in Figure 3-29.

```
SELECT DISTINCT APPLID,TRANSID,PROGRAM,FUNCTION
FROM CICSIA53.CIU_CICS_DATA
WHERE FUNCTION='DISPLAY'
ORDER BY APPLID, TRANSID, PROGRAM, FUNCTION
```

Figure 3-29 Pasted SQL command

Comparing the results

The Show Resources view can be used to compare the results of two similar queries. In the following example, we compare resources used by region IYDZZ518 and region IYDZZ528. We created and executed two queries that show the resources used by the respective regions. We can then select the compare icon as shown in Figure 3-30.

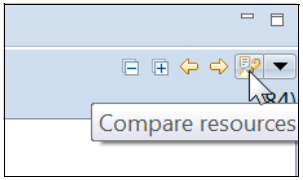


Figure 3-30 Compare query results

You can then select the results that you want to compare as shown in Figure 3-31.

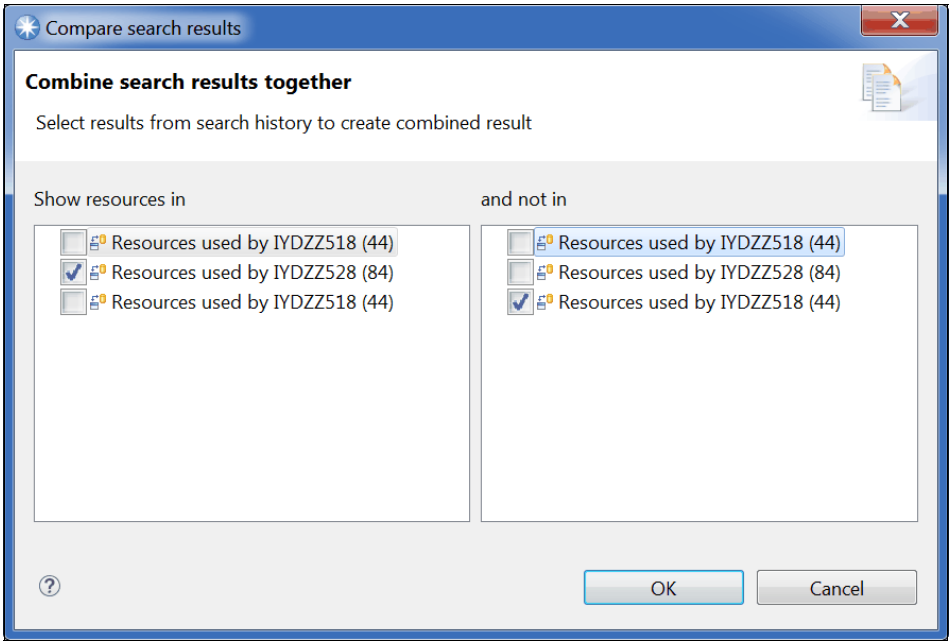


Figure 3-31 Compare resources in IYDZZ528 and not in IYDZZ518

The results are shown in the Show Resources view as shown in Figure 3-32. We can see that transaction SSP4 is only used in CICS region IYDZZ528.

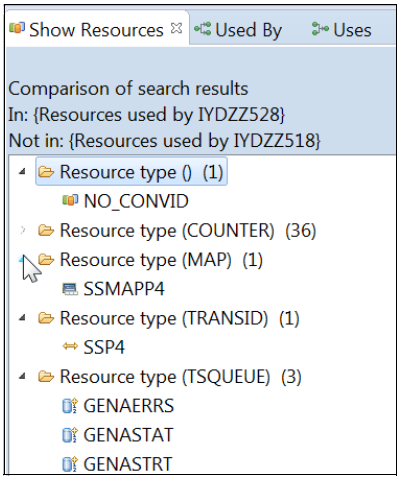


Figure 3-32 Compare results

4

Using CICS IA data for affinity analysis

In this chapter, we provide information about how CICS Interdependency Analyzer (IA) can assist in discovering CICS Transaction Server (TS) affinities, which you need to understand before you embark on balancing your work load across cloned CICS regions.

Enabling workload balancing enables you to dynamically route transactions and programs across your CICS regions. The following main benefits come from practicing workload balancing:

- Performance
- Availability
- Scalability

The following topics are covered:

4.1 What are affinities?

In this section, we describe what affinities are, look at the different types of affinities, describe affinity relations, and life times. For more information about affinities, see the CICS TS IBM Knowledge Center at the following link:

https://www.ibm.com/support/knowledgecenter/SSGMCP_5.2.0/com.ibm.cics.proddoc.doc/topics/KC.html

4.1.1 Understanding affinities

CICS transactions use many different techniques to pass data from one to another. Some techniques require that the transactions exchanging data must execute in the same CICS region, and therefore impose restrictions on the dynamic routing of transaction. If transactions exchange data in ways that impose such restrictions, there is said to be an **affinity** between them.

There are two categories of affinities.

Inter-transaction affinity

An *inter-transaction affinity* is an affinity between two or more CICS transactions. It is caused by the transactions using techniques to pass information between one another, or to synchronize activity between one another, in a way that requires the transactions to execute in the same CICS region. Inter-transaction affinity, which imposes restrictions on the dynamic routing of transactions, can occur in the following circumstances:

- One transaction terminates, leaving “state data” in a place that a second transaction can access only by running in the same CICS region as the first transaction.
- One transaction creates data that a second transaction accesses while the first transaction is still running. For this process to work safely, the first transaction usually waits on some event, which the second transaction posts when it has read the data created by the first transaction. This synchronization technique requires that both transactions are routed to the same CICS region.

Transaction-system affinity

A *transaction-system affinity* is an affinity between a transaction and a particular CICS region (that is, it is not an affinity between transactions themselves). It is caused by the transaction interrogating or changing the properties of that CICS region. Transactions with affinity to a particular system, rather than to another transaction, are not eligible for dynamic transaction routing. In general, they are transactions that use **INQUIRE** and **SET** commands, or depend on global user exit programs.

The restrictions on dynamic routing caused by transaction affinities depend on the duration and scope of the affinities. Clearly, the ideal situation for a dynamic routing program is for there to be no transaction affinity at all, which means there is no restriction in the choice of available target regions for dynamic routing. However, even when transaction affinities do exist, there are limits to the scope of these affinities determined by the:

- Affinity lifetime
- Affinity relation

Affinity lifetime

The *affinity lifetime* determines when the affinity is ended. An affinity lifetime can be classified as one of the following types.

System

The affinity lasts while the target region exists, and ends whenever the target region terminates (at a normal, immediate, or abnormal termination). (The resource shared by transactions that take part in the affinity is not recoverable across CICS restarts.)

Permanent

The affinity extends across all CICS restarts. (The resource shared by transactions that take part in the affinity is recoverable across CICS restarts.) This type of affinity lifetime is the most restrictive of all the inter-transaction affinities.

Process

The affinity exists until the process completes.

Activity

The affinity exists until the activity completes.

Pseudo-conversation

The (LU-name or user ID) affinity lasts for the whole pseudo-conversation, and ends when the pseudo-conversation ends at the terminal.

Logon

The (LU-name) affinity lasts while the terminal remains logged on to CICS, and ends when the terminal logs off.

Signon

The (user ID) affinity lasts while the user is signed on, and ends when the user signs off.

Affinity relation

The *affinity relation* determines how the dynamic routing program selects a target region for a transaction instance associated with the affinity. An affinity relation can be classified as one of the following.

Global

A group of transactions where all instances of all transactions in the group that are initiated from any terminal must execute in the same target region for the lifetime of the affinity. The affinity lifetime for global relations can be system or permanent.

By all instances, we mean that a transaction started from a terminal, by the **START** command or a BTS process.

BAPPL

All instances of all transactions in the group are associated with the same CICS Business Transaction Services (BTS) process. There may be many different user IDs and terminals associated with the transactions included in this affinity group.

LU-name

A group of transactions where all instances of all transactions in the group that are initiated from the same terminal must execute in the same target region for the lifetime of the affinity. The affinity lifetime for LU-name relations can be pseudo-conversation, logon, system, or permanent.

User ID

A group of transactions where all instances of the transactions that are initiated from a terminal and executed on behalf of the same user ID must execute in the same target region for the lifetime of the affinity. The affinity lifetime for user ID relations can be pseudo-conversation, signon, system, or permanent.

Notes:

For user ID affinities, the pseudo-conversation and signon lifetimes are possible only in those situations where one user per user ID is permitted. Such lifetimes are meaningless if multiple users are permitted to be signed on with the same user ID at the same time (at different terminals).

If an affinity is both user ID and LU-name (that is, all instances of all transactions in the group were initiated from the same terminal and by the same user ID), LU-name takes precedence.

4.1.2 Transaction system affinities

In this section, we look at the different types of affinities. The CICS IA affinity collector captures all of these types.

ENQ/DEQ

The affinity here is between all transactions that issue an EXEC CICS ENQ and DEQ on a resource.

The resource can be:

- Character string of length 1 - 255 bytes
- Address

The affinity relation can be:

- GLOBAL
- USERID
- BAPPL

The lifetime is always SYSTEM.

Note:

Commands that result in LENGTHERR condition are grouped together and treated as a resource name of ‘LENGERR.’

All other errors are treated as normal and the resource is collected.

READQ TS, WRITEQ TS, DELETEQ TS

The affinity here is between all transactions that use the same TS queue to both MAIN and AUXILIARY TSQueues. The match is made on the name of the TS queue.

The affinity relation can be one of the following names:

- GLOBAL
- USERID
- LUNAME
- BAPPL

The lifetime can be one of the following names:

- PCONV
- LOGON
- SIGNON
- ACTIVITY
- PROCESS
- SYSTEM
- PERMANENT

Note:

- MAIN TS Queue cannot be recovered so cannot be PERMANENT.
- No data is collected if it is defined as remote or if a remote SYSID is specified on the command.
- Data is collected for commands in error.
- If the TS queue is created and deleted within the same task, no data is collected.
- Scanner detects all instances of TS commands.

ADDRESS CWA

The affinity here is between all transactions that issue ADDRESS CWA.

The affinity relation can be:

- GLOBAL
- BAPPL

The lifetime is always SYSTEM.

LOAD HOLD/RELEASE

The affinity here is between all transactions that LOAD HOLD and RELEASE the same program (or TABLE). The match is made on the program name.

The affinity relation can be:

- GLOBAL
- BAPPL

The lifetime is always SYSTEM.

Note:

- LOAD and RELEASE protocol applies only to programs that are defined with RELOAD(NO). If the collector cannot establish the RELOAD attribute for some reason, RELOAD(NO) is assumed.
- When a LOAD HOLD has occurred for a program, any subsequent LOAD (with or without HOLD) or RELEASE is part of the affinity.
- LOAD with no HOLD for programs defined as RESIDENT is not treated as an affinity.

  Relying on residency for sharing is inherently unsafe. The program can be replaced by SET PROG NEWCOPY.
- Incorrect use of RELEASE for a program defined with RELOAD(YES) is not detected.
- Data is collected for commands in error.
- Scanner detects all instances of LOAD, not just LOAD HOLD, and all instances of RELEASE.

GETMAIN SHARED/FREEMAIN

The affinity here is between the transaction that obtains storage via GETMAIN SHARED and the transaction that frees the same piece of storage via FREEMAIN. Both transactions must be seen for there to be an affinity. The match is made on the storage address.

Note: The situation is complicated by the fact that the storage address might be passed to other transactions, and if they access the storage, they cannot be detected because the access is not through a CICS API.

The affinity relation can be:

- GLOBAL
- BAPPL
- LUNAME
- USERID

The lifetime can be:

- PCONV
- LOGON
- SIGNON
- ACTIVITY

- PROCESS
- SYSTEM

Note:

- The detector always worsens LOGON and SIGNON to SYSTEM because of limitations in the way it is collected.
- Commands in error are ignored because there is no address for matching GETMAIN with FREEMAIN, no data is collected.
- A GETMAIN/FREEMAIN affinity is considered to be initiated from a terminal if the GETMAIN is initiated from a terminal, whether the FREEMAIN was so initiated or not is irrelevant.
- Unmatched GETMAIN SHAREDs are also reported if they have never matched by the time the detector is stopped.
- Scanner detects all instances of GETAMIN SHARED and all instances of FREEMAIN.

RETRIEVE WAIT/START

The affinity here is between all transactions that issue START commands for a particular transaction at a terminal, where the started transaction issues RETRIEVE WAIT. The transaction that issues the RETRIEVE WAIT is also part of the affinity. The match is made on the transid.

The affinity relation can be:

- GLOBAL
- USERID

The lifetime can be:

- SYSTEM
- PERMANENT

Note:

- Lifetime is PERMANENT is assumed if PROTECT is specified on the START.
- If the transaction to be started is defined as remote or a remote SYSID was specified on the START command (function shipped), no data is collected.
- Data is collected for commands in error.
- Scanner detects all instances of RETRIEVE WAIT, and all instances of START that either specify TERMID, or omit NOCHECK, or specify REQID (because of CANCEL affinity).

LOAD /FREEMAIN

The affinity here is between the transaction that loads the program via LOAD and the transaction that releases the same program via FREEMAIN. The match is made on the load point address.

Note: The situation is complicated by the fact that the load point address might be passed to and used by other transactions.

The affinity relation can be one of the following names:

- GLOBAL
- LUNAME
- USERID
- BAPPL

The lifetime can be one of the following names:

- PCONV
- LOGON
- SIGNON
- ACTIVITY
- PROCESS
- SYSTEM

Note:

- The detector always worsens LOGON and SIGNON to SYSTEM because of limitations in the way it is collected:
  - LOAD and FREEMAIN protocol applies only to programs that are defined as RELOAD(YES).
  - HOLD is irrelevant because the CICS Program Control never sees the FREEMAIN or knows the storage location of the individual task's copy, and so it cannot release the program at task end.
- The above implies that all LOADs must be examined as they are all effectively LOAD HOLDs.
- Commands in error are ignored because there is no load address for matching the LOAD with FREEMAIN so no data is collected.
- LOAD with no SET option is ignored because no address is returned.
- A LOAD/FREEMAIN affinity is considered to be initiated from a terminal if the LOAD is initiated from a terminal, whether the FREEMAIN was so initiated or not is irrelevant.
- Unmatched LOADs are also reported if they have never matched by the time the detector is stopped.
- Scanner detects all instances of LOAD and all instances of FREEMAIN.

CANCEL/DELAY/POST/START

The affinity here is between the transaction that issues the **DELAY**, **POST**, or **START** command and the transaction that issues the **CANCEL** command via REQID. The match is on the REQID.

In order for another task to CANCEL a DELAY, the REQID must be explicitly specified on the **DELAY** command. If no REQID is specified, it cannot be canceled and therefore cannot be detected.

In order for another task to CANCEL a START or POST, it is not necessary to specify REQID on the command because CICS supplies a unique REQID that can be used (unless START specifies NOCHECK). So only START commands that do not specify NOCHECK and omit REQID, and all POST commands are detected.

The affinity relation for START can be one of the following names:

- GLOBAL
- BAPPL
- LUNAME
- USERID

The affinity relation for DELAY and POST can be one of the following names:

- GLOBAL
- BAPPL
- LUNAME
- USERID

The lifetime for START can be one of the following names:

- PCONV
- LOGON
- SIGNON
- ACTIVITY
- PROCESS
- SYSTEM
- PERMANENT

Note:

- The PROTECT option determines whether SYSTEM or PERMANENT is used.
- LOGON and SIGNON always worsened to SYSTEM or PERMANENT.

The lifetime for DELAY and POST can only be one of the following names:

- SYSTEM
- PROCESS
- ACTIVITY
- PCONV

Note:

- If the relation is LUNAME or USERID, lifetime is PCONV because neither DELAY or POST exists beyond task termination.
- If the transaction specified on a START or CANCEL command is defined as remote, or a remote SYSID was specified on the command (function shipped), no data is collected.
- If we cannot establish the RELOAD attribute for some reason, RELOAD(NO) is assumed.
- A CANCEL affinity is considered to be initiated from a terminal if the START, DELAY, or POST is initiated from a terminal, whether the CANCEL was so initiated or not is irrelevant.
- Scanner detects all instances of POST, all instances of DELAY REQID, all instances of CANCEL REQID, and all instances of START that either omit NOCHECK or specify REQID or specify TERMID (because of the RETRIEVE WAIT affinity).
- START, DELAY, and POST commands in error are ignored, so no data is collected.
- Data is not collected for commands that expire on entry into interval control because they cannot be canceled.
- CANCEL commands that omit REQID are ignored because they cannot cancel another task.
- CANCEL commands that return a NOTFND response are ignored because the *interval control element* (ICE) must have expired.
- REQIDs are assumed to be unique, that is, there are no simultaneous pairs of START/CANCEL using the same REQID.
  - Violates CICS programming guidelines
  - Results from CICS are unpredictable

4.1.3 Inter-transaction affinities

The commands involved with inter-transaction affinities are:

- ENABLE/DISABLE PROGRAM
- EXTRACT EXIT
- INQUIRE
- SET
- PERFORM
- RESYNC
- DISCARD

- CREATE
- WAIT EXTERNAL
- WAIT EVENT
- WAITCICS
- COLLECT STATISTICS
- CSD commands

Note:

- The affinity here is not an affinity between transactions, but rather an affinity with the system on which the command was issued.
- Such affinities do not generate transaction affinity groups because it does not make sense to dynamically route such transactions.
- The use of these commands does require reporting because the system programmer should be aware of transactions and programs that issue such commands.
- No scanner differences.

4.2 Collecting and loading affinity data

As we learned in the previous section, there are several types of affinities. The CICS IA CINT transaction allows you to select which type you want to collect.

To get the best understanding of our affinities, we need to collect the best data. The best data can be collected in your CICS production regions. The collection of such data can introduce performance considerations. Some of the preferred practices for collecting CICS IA data are discussed in 2.1, “Collecting CICS IA data in a production region” on page 24. However, for affinity collection, we cannot use the new optimum collection option described in 2.1.1, “Using the Optimum Collection option” on page 24.

In order to avoid performance issues when collecting affinity data, it is recommended that you collect each type of affinity separately. For example, in the first week, collect CWA affinities. Then in the second week, collect ENQ/DEQ affinities while at the same time loading and analyzing your CWA affinities.

4.2.1 Configuring affinity options

To modify and save your affinity collection, you need to use the CICS CINT transaction. Select option 2 to “Configure Region Options”, then select 4 for “Options” against the DEFAULT entry or a specific REGION entry. This selection takes you to the Resource Options menu as shown in Figure 4-1.

For more information about the CINT transaction, see the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html

To collect affinity data, we need to set the “Data to Collect” flag to “A”.

```
Data to Collect  . . . . . : A    (A=Affinity, I=Interdependency, B=Both)
```

```
CIU290        CICS Interdependency Analyzer for z/OS - V5R3M0       2015/09/09
                      Resource Options for                          10:56:45AM
                CICS Sysid:  DFTS  CICS Applid:  DEFAULTS

Type action code then press ENTER: 7

    1 = General Options
    2 = Time/Date Options

    Interdependency Options            Affinity Options
    3 = CICS Options for APIs          7 = Affinity Options
    4 = CICS Options for SPIs
    5 = DB2/IMS/MQ/CPSM Options        CICS TS Applications
    6 = Natural Options                8 = Application Data



CICS Sysid: Z518    CICS Applid: IYDZZ518    TermID: TC42

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Exit
```

Figure 4-1 CINT: Resource options

This option is found under option 1 for “General Options” as shown in Figure 4-2 on page 95.

```
CIU26O        CICS Interdependency Analyzer for z/OS - V5R3MO        2015/09/09
                          General Options for                        11:08:06AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS
Modify the options and press Enter to update, or PF12 to Cancel.
Control options
  Data to Collect  . . . . . : A    (A=Affinity, I=Interdependency, B=Both)
  Perform periodic saves . . . : N    (Y=Yes, N=No)
  Trigger for CINB start . . . : 1    (2 to 9999 thousand record updates)
  Restore data on start  . . . : N    (Y=Yes, N=No)
  Maintain usage counts  . . . : Y    (Y=Yes, N=No)
  Size of dataspace  . . . . . : 16   (1O to 2OOO Mbytes)
  Transid prefix (optional). . :      (1 to 4 characters)
  Program exclude list . . . . : CIUXPROG (1 to 8 characters)
  Transaction exclude list . . : CIUXTRAN (1 to 8 characters)
  Command exclude list . . . . : CIUXCOMM (1 to 8 characters)
  Resource compression list. . : CIUXRCOM (1 to 8 characters)
  Dump HLQ . . . . . : DUMP     (1 to 8 characters)
  Dynamic call . . . . . : Y    (Y=Yes, N=No)
  Trigger for Task collection. : 1    (1 to 9999)
  Collect Long Running Tasks . : N    (Y=Yes, N=No)
CICS Sysid:  Z518   CICS Applid:  IYDZZ518   TermID:  TC42

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F1O=         F11=         F12=Cancel
```

Figure 4-2 CINT: General options

To select which types of affinities that we want to collect then, we need to go to the Affinity Options menu, which you can get to by selecting option 7 in the Region Options menu, as shown in Figure 4-1 on page 94.

The Affinity Options menu, shown in Figure 4-3, allows you to select which types of affinity that you want to collect.

```
CIU27O   CICS Interdependency Analyzer for z/OS - V5R3MO            2015/09/09
              CICS Affinities Options for                           11:20:06AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.

Detect affinity types: Y=Yes, N=No
                       T=Terminal associated task (TS QUEUE type only)

Inter-Transaction
 ENQ, DEQ . . . Y  TS QUEUE . . .  Y  ADDRESS CWA. . Y RETRIEVE WAIT. Y
 LOAD . . . . . Y  GETMAIN SHARED  Y  CANCEL . . . . Y

Transaction-System
 INQUIRE, SET . Y  ENABLE, DISABLE Y  EXTRACT. . . . Y COLLECT STATS. Y
 PERFORM  . . . Y  RESYNC . . . .  Y  WAIT . . . . . Y DISCARD  . . . Y
 CREATE . . . . Y  CSD  . . . . .  Y

Multiple signon with same ID : N    (Y=Yes, N=No)

CICS Sysid:  Z518   CICS Applid:  IYDZZ518   TermID:  TC42

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F1O=         F11=         F12=Cancel
```

Figure 4-3 CINT: Affinity Options menu

It is recommended that if you are collecting in production, you should collect all inter-transaction affinity types at separate times. Then, to collect all of the Transaction-System affinities in one run, start the collector with all Inter-Transaction affinity flags switched off. The more options you select, the more CICS IA exits are enabled and the greater the impact on the performance.

As you can see in Figure 4-3, there is also one more option that you need to set:

`Multiple signon with same ID : N    (Y=Yes, N=No)`

If your site allows users to sign on to the CICS region multiple times, set this option to Y for YES. This option is used when you calculate the affinity relation and lifetime.

We are now ready to start the collector and load the data into the DB2 tables.

4.2.2 Running the affinity collector and loading the data

You can start the IA collector in many ways:

- By using the CINT transaction. See Figure 4-4.
- By using the plug-in: See Figure 4-5.
- From the CICS console: See Example 4-1 on page 98.
- During CICS PLT startup: See Example 4-2 on page 98.

```
CIU10O        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/09
                              Operations Menu                        11:33:01AM

Type action code then press ENTER.                                    More :

1= Start 2= Stop 3= Pause 4= Continue 5= Statistics 6= Refresh Run Options

      CICS      CICS               Start       Start
 Act  Applid    Sysid Status       Date        Time        Collecting
      ALL       ALL
  1   IYDZZ518  Z518  STOPPED
      IYDZZ528  Z528  STOPPED
      IYDZZ538  Z538  UNCONNECTED



CICS Sysid:  Z518   CICS Applid:  IYDZZ518   TermID:  TC42

F1=Help      F2=           F3=End       F4=          F5=Refresh  F6=
F7=Page Up   F8=Page Down  F9=          F10=         F11=         F12=
```

Figure 4-4 CINT: Operations menu

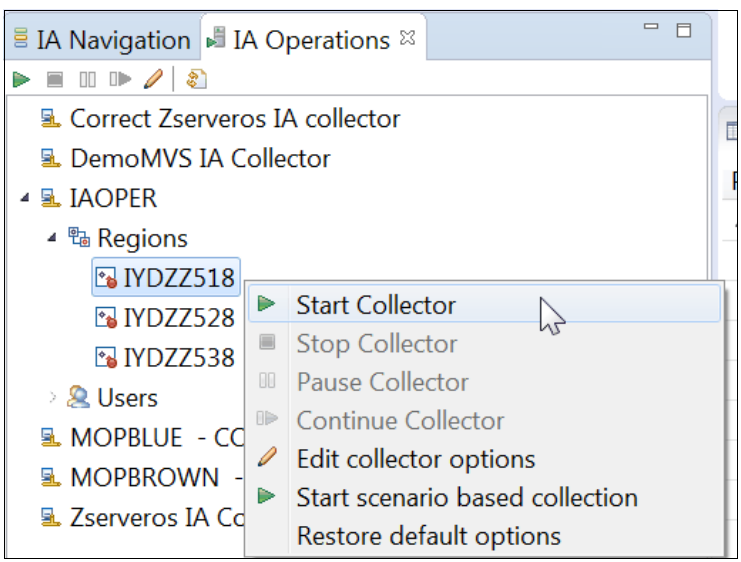


Figure 4-5 Starting the collector from the plug-in

Example 4-1 Operating the collector from a CICS console

```
/F cisappl,CINT START

/F cicsappl,CINT STOP
```

Example 4-2 Sample code for PLT startup

```
A PLTPI program to start the Collector, CIUSTART, is supplied with CICS
IA. To start CICS IA from the PLT, add the following lines to your PLT
startup table:

DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT TYPE=ENTRY,PROGRAM=CIUSTART
```

After you start the collector, you can view the CIULOG DD for the CICS region to see what options are active and what exits are enabled, as shown in Figure 4-6 on page 99. You can see that we have selected to collect all affinity options and that there are five exits enabled.

```
CINT START is requested by CICSUSER for IYDZZ518
EXEC CICS ENABLE EXIT for program CIUXDUMM, EXIT XICEXP OK
CINT Collector runtime options for IYDZZ518:
GLOBAL OPTIONS:
........
GENERAL OPTIONS:
    DATA_TO_COLLECT=A PERIODIC_SAVES=N CINB_TRIGGER=1
.......
DATE/TIME OPTIONS:
.......
DEPENDENCY OPTIONS:
.......
AFFINITY OPTIONS.
     INTER-TRANSACTION:
       ENQ/DEQ=Y TS_QUEUE=Y ADDRESS_CWA=Y RETRIEVE_WAIT=Y LOAD=Y
       GETMAIN_SHARED=Y CANCEL=Y
     TRANSACTION-SYSTEM:
       INQUIRE/SET=Y ENABLE/DISABLE=Y EXTRACT=Y COLLECT_STATS=Y
       PERFORM=Y RESYNC=Y WAIT=Y DISCARD=Y CREATE=Y CSD=Y
     MULTIPLE_SIGNON=N
 EXEC CICS ENABLE EXIT for program CIUZCCO1, EXIT XEIOUT OK
 EXEC CICS ENABLE EXIT for program CIUZCCX1, EXIT XICEXP OK
 EXEC CICS ENABLE EXIT for program CIUZCCM1, EXIT XMEOUT OK
 EXEC CICS ENABLE EXIT for program CIUZCCB1, EXIT XBADEACT OK
 EXEC CICS ENABLE EXIT for program CIUZCCF1, EXIT XFAINTU OK
 EXEC CICS ENABLE TRUE for program CIUZCCI1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCO1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCX1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCM1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCB1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCF1 OK
 EXEC CICS ENABLE EXIT START for program CIUZCCI1 OK
 Affinity collector is now RUNNING
 Affinity files are emptied
Transaction CINS is starting
CINB task is starting
```

Figure 4-6 Sample CIULOG

When we collected our affinity data, we need to stop the collector and load the data into the DB2 data.

Again, you can stop the IA collector in many ways:

- By using the CINT transaction: See Figure 4-4 on page 97.
- By using the plug-in: See Figure 4-5 on page 97.
- From the CICS console: See Example 4-1 on page 98.
- During CICS PLT shutdown.

Loading the affinity data

Now that we have collected our affinity data, we need to load it into the DB2 data tables so that we can analyze the data.

You need to edit and run the SCIUSAMP(CIUAFFLD) job to load the affinity from the VSAM file into the DB2 tables. The first step in this job STEP000 updates the CIU_REGION_INFO table with a time stamp of when the load was done and we pass it a collection ID. You can set this collection ID, _collid_, to any 16 character.

Note: For affinities, the collection ID is only stored on the CIU_REGION_INFO row. Unlike the dependency data, it is not attached to the actual affinity collection tables.

The second step, STEP010, processes the affinity VSAM files and creates an output file containing affinity “events”, which is sorted and then loaded into the CIU_AFF_EVENTS table using a LOAD utility. Example output from STEP010 is shown in Figure 4-7.

```
CIU6011I  ****  QSAM OUTPUT STATISTICS FOR TABLE CIU_AFF_EVENTS ****
CIU6012I  ---------------------------------------------
CIU6053I  Data records read:      000000052
CIU6054I  Data records processed: 000000052
CIU6055I  Data records skipped:   000000000
```

Figure 4-7 Sample output for step STEP010 in CIUPRINT

Program CIUAFFL1 is called in STEP040 to process the rows in the CIU_AFF_EVENTS table and update the three affinity tables:

- CIU_AFF_GRP_DATA
- CIU_AFF_CMD_DATA
- CIU_AFF_INDEX

In this step, you can choose to process the affinities for a specific CICS region by using the //APPLID DD card as shown in Figure 4-8.

```
//* Enter ALL for all APPLIDs or the APPLID to be loaded
//* No entry is treated the same as ALL
//APPLID   DD   *
/*
```

Figure 4-8 Select your CICS region

We are now ready to run the affinity report and builder sample jobs. You can do this task in batch by running the following jobs:

SCIUSAMP(CIUAFFRD) This job produces the affinity report and input into the builder job.

SCIUSAMP(CIUAFFBL) This job takes input from the report job and outputs CPSM transaction groups that CICS TS CPSM can consume when you define your work load management policy.

In this book, we use the CICS IA plug-in to run the reporter and the builder. For more information about the batch jobs, see the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

4.2.3 Running the affinity report from the IA plug-in

In this section, we run an affinity report from the CICS IA Explorer plug-in for CICS region IYDZZ518.

The affinity report is run against CICS regions, so first we need to open the `Regions` folder in the IA Navigation view and then select **Report → Affinity Report** as shown in Figure 4-9.

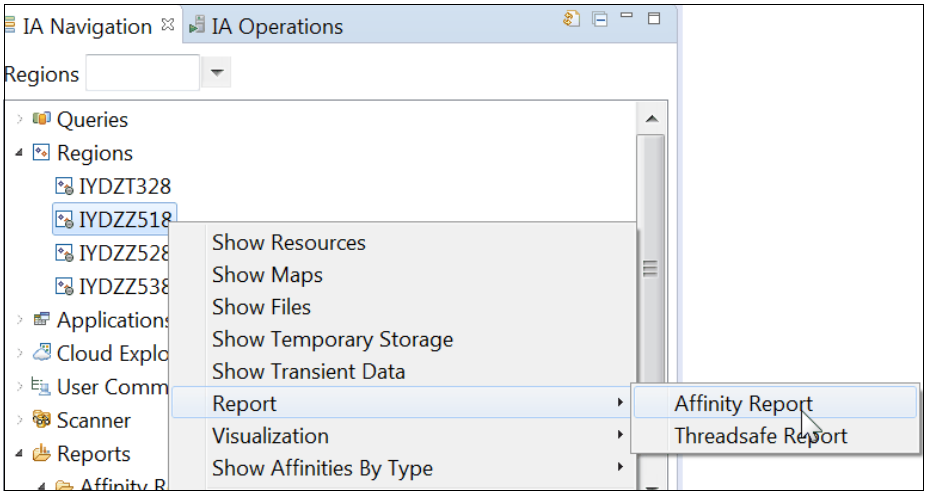


Figure 4-9 Run an affinity report against region IYDZZ518

This action drives the affinity report wizard. We now go through the steps required to complete the report. Figure 4-10 shows the first step of creating your report. It allows you to add and remove the CICS regions for which you want to report and also the type of affinity you want to include in the report.

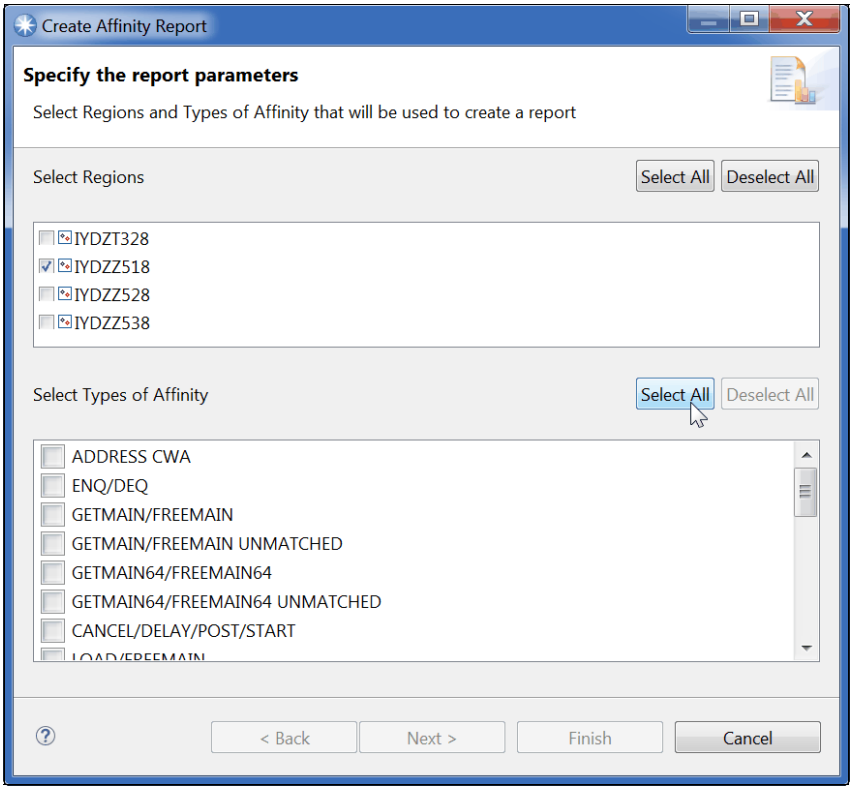


Figure 4-10 Select your CICS regions and affinity types

As mentioned earlier, for performance reasons we recommend processing the different types of affinities separately. The affinity report wizard allows you to generate reports for each type of affinity. Later, we see how we can select all the different type of reports as input into the affinity builder, which is covered in 4.2.4, “Running the affinity builder” on page 104.

In this example, we select to generate a report for the ENQ/DEQ affinity. The next step is to select a location where to save the report and to give it a meaningful name as shown in Figure 4-11 on page 103.

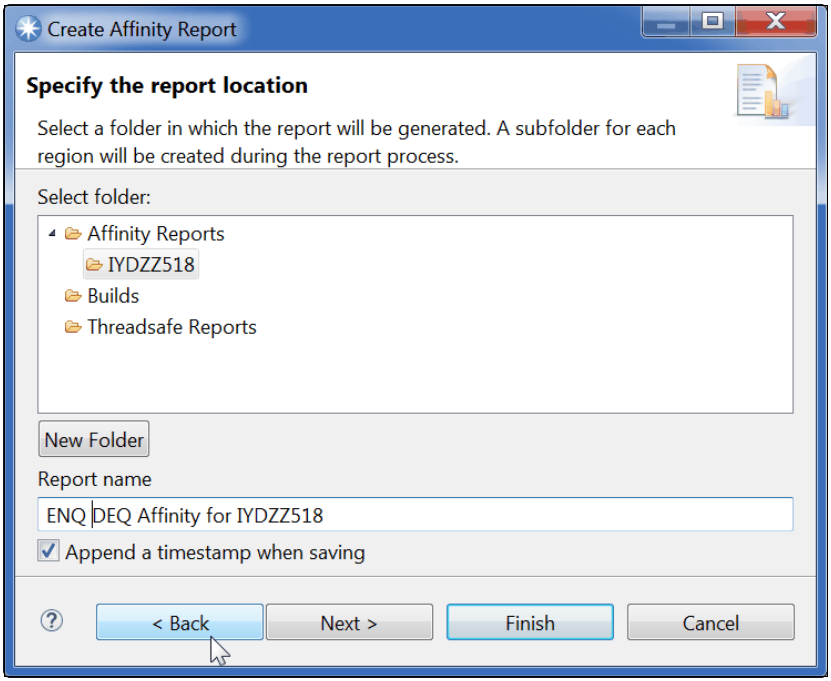


Figure 4-11 Save your report under your CICS region and name your report

You can also choose to append a time stamp when saving the report. You can then click **Finish** to generate the report shown in Figure 4-12 on page 104.

The report view is separated into two folders:

- Overview
- Transaction Groups

The overview is divided into two parts. On the left side, we can see the affinity groups. A group is created for each different resource name. In this case, we have seen an affinity for three different ENQ/DEQ resources. For each group, we also display the affinity relation and the affinity lifetime. By selecting a group you can populate the right side, which lists the commands that cause the affinity by transaction and program.

In this example, we can see that the first group is an ENQ/DEQ affinity and we can see the transaction AFFS issues an ENQUEUE and a DEQUEUE on this resource.

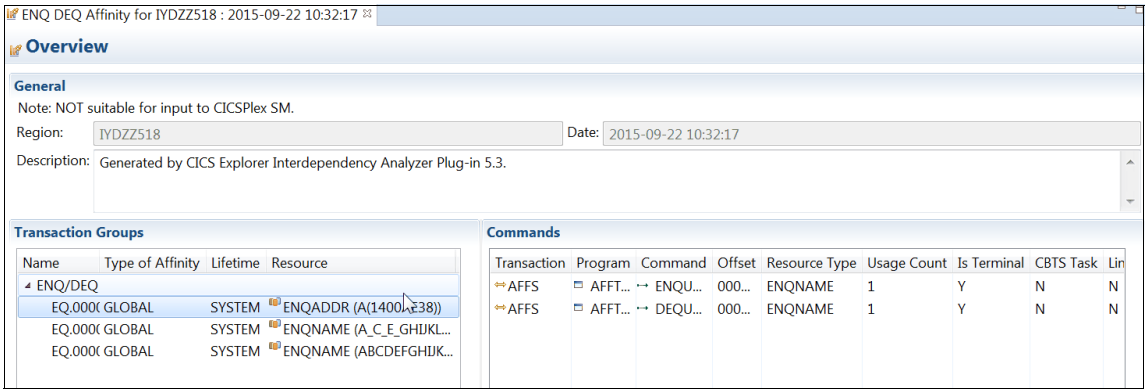


Figure 4-12 ENQ/DEQ affinity for region IYDZZ518

The `Transactions Group` folder shown in Figure 4-13 lists the CPSM WLM transaction groups that could be created by the CICS IA Builder.

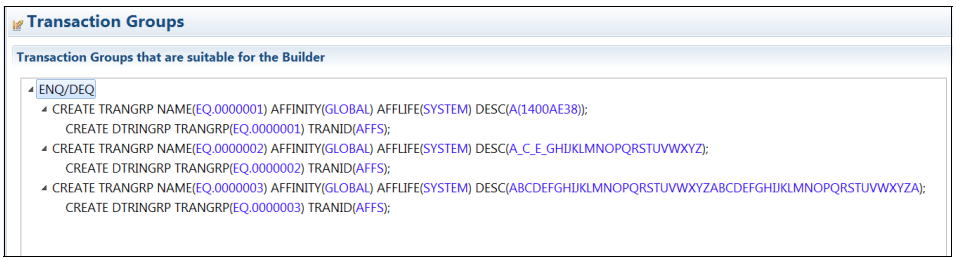


Figure 4-13 Possible CPSM WLM affinity groups

You can now collect data and run reports for other types of affinities.

4.2.4 Running the affinity builder

After you have collected and ran reports for all affinity types, you can then choose to remove those affinities or use the affinity builder to create CPSM Transaction Groups that can be deployed against a CPSM WLM definition.

The CICS IA Builder can also be ran as a batch job to build affinity transaction group definitions suitable for input to the CICS system management product, IBM CICSPlex SM.

As mentioned earlier, you can use the CICS IA affinities reporter to produce files of basic transaction affinity groups for input to the builder. The reports that we created previously are saved in the `Reports` folder in the IA Navigation view.

Select one of your reports, right-click, and select “Build CPSM WLM Transaction Groups” as shown in Figure 4-14.

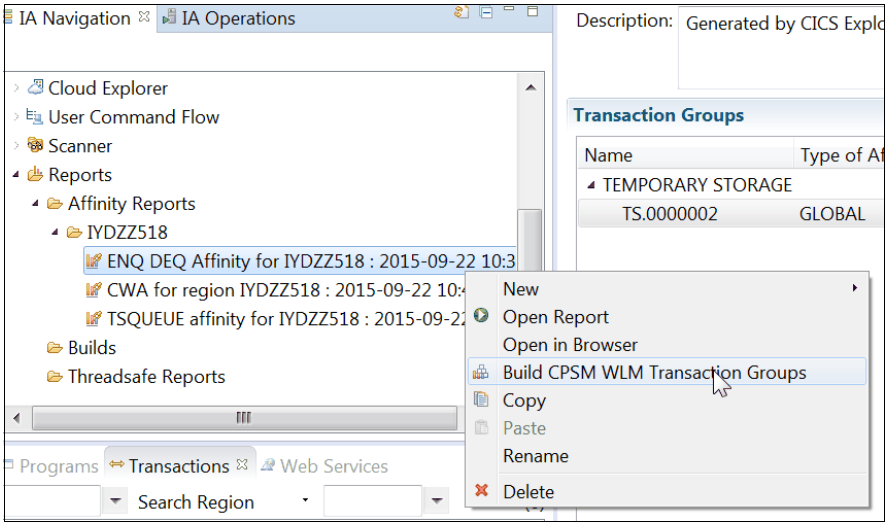


Figure 4-14 Build CPSM WLM Transaction Groups

This action drives the IA Builder wizard. We now go through the steps to create the builder output. The first step shown in Figure 4-15 on page 106, allows you to select the affinity reports to input into the IA Builder wizard. You can also select previous IA Builder output as input to the IA Builder wizard.

The IA Builder wizard takes as input a set of files containing basic affinity transaction groups, combines those groups, and produces a file containing combined affinity transaction groups. CICSPlex SM requires a transaction identifier to be in one transaction group only, and the IA Builder satisfies this requirement by combining groups that contain the same transaction identifier.

We select all the reports for region IYDZZ518 and click **Finish** to generate the builder output. In this section, you also need to provide the following.

CONTEXT

Specify the name, 1 -8 characters, of a CICSplex. If you specify this parameter, the Builder generates a CICSPlex SM CONTEXT statement, which enables CICSPlex SM to associate the combined affinity transaction groups with a particular CICSplex that it is managing. The default is to not generate a CONTEXT statement; in which case, CICSPlex SM assumes the local CICS managed address space (CMAS).

MATCH: LUNAME or USERID

Specify the filter that CICSPlex SM will use for workload separation, and which applies to all combined affinity groups produced by the Builder.

STATE: ACTIVE or DORMANT

Specify whether the combined affinity groups are to be defined as active or dormant to CICSPlex SM.

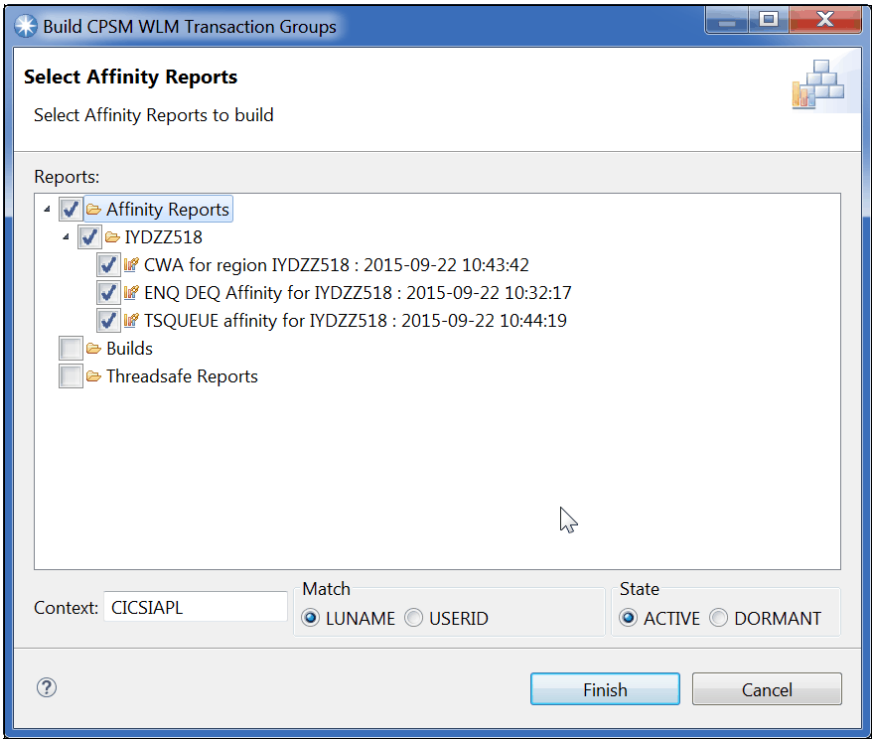


Figure 4-15 Select affinity reports for region IYDZZ518

After pressing **Finish**, you are presented with a new Builder output. This view consists of three folders:

- Overview
- CPSM Input
- Deployment JCL

We now look at these three folders in turn.

Builder overview

The overview shows the options that we selected and a preview of the transaction groups that will be created as shown in Figure 4-16.

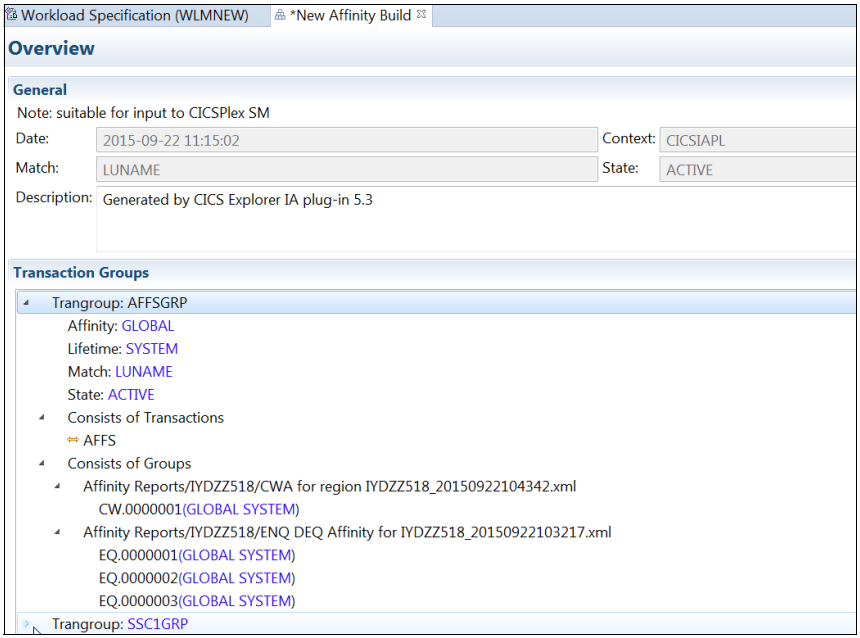


Figure 4-16 Builder overview

Builder CPSM input

This folder allows you to enter the CPSM input parameters required to build the JCL that is used to create the CPSM transaction groups. As shown in Figure 4-17 on page 108, you need to input the following names:

- **Data set name and member name:** This input parameter is the IBM MVS™ dataset name where the actual CPSM CREATE definitions are stored.
- **CMAS name:** Specifies the 1 - 8 character name of a CMAS to which the job is to connect.
- **Output user name:** Specifies a 1 - 8 character user ID identifier to be associated with the spooled output.
- **Print node:** Specifies a 1 - 8 character print node identifier to be used by the system spooler for routing the job output.
- **CHECK or EXECUTE:** Performs a syntax check or executes the **`CREATE`** command.

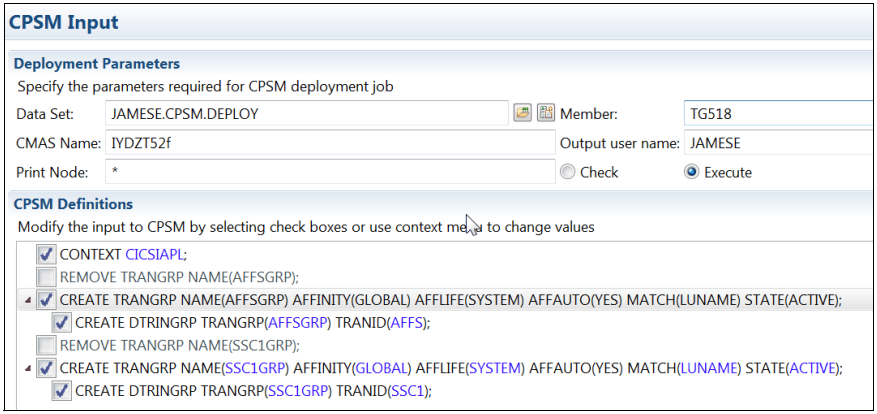


Figure 4-17 Enter your CPSM input

The CPSM Definitions in Figure 4-17 allow you to select which groups to create. It also allows you to remove previously created groups.

If you have a z/OS FTP connection or a z/OSMF connection to your host, you can use the icons shown in Figure 4-18 and Figure 4-19 to open an existing data set or create a new one, respectively.

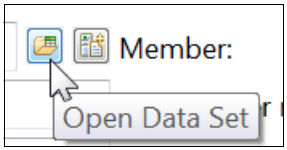


Figure 4-18 Open Data Set icon

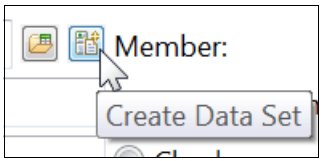


Figure 4-19 Create Data Set icon

Deployment JCL

Before you can use this JCL, you need to configure some IA settings to create a default JOBCARD and allow you to specify the CPSM High-Level Qualifiers. To do the deployment, select **Window → Preferences** from the toolbar as shown in Figure 4-20 on page 109.

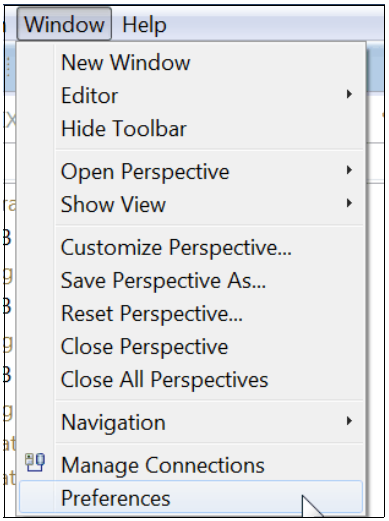


Figure 4-20 Set preferences

Then, select the CPSM Deployment settings under CICS IA as shown in Figure 4-21. Enter your default job card and the CPSM library names.

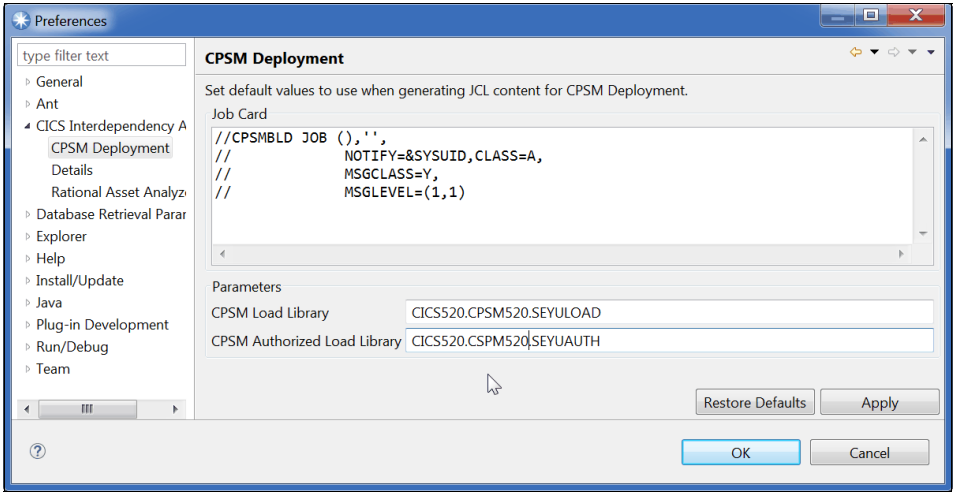


Figure 4-21 CICS IA CPSM Deployment settings

The `Deployment JCL` folder shown in Figure 4-22 allows you to review the JCL before submitting the job by selecting the green “go” arrow in the upper-right corner. In order to submit the job, you need a z/OS FTP connection or a z/OSMF connection to the host.

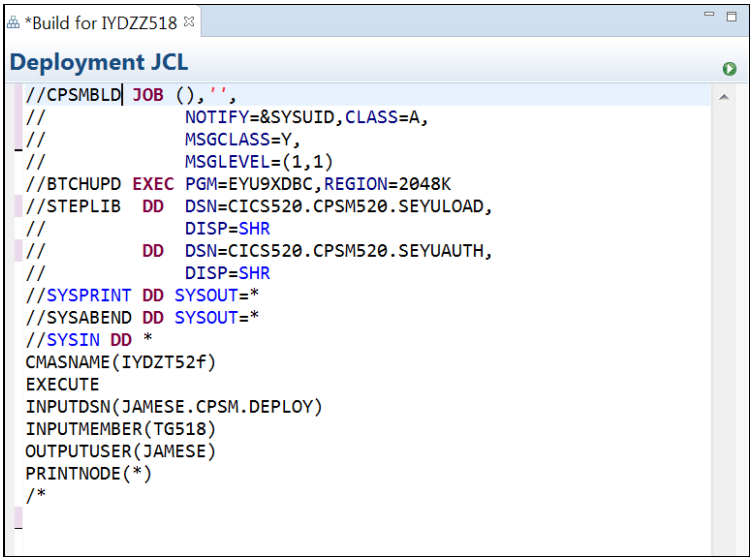


Figure 4-22 JCL to generate the CPSM transaction groups

After you submit the job, the Console view opens as shown in Figure 4-23.

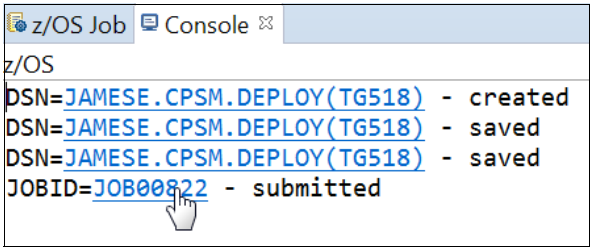


Figure 4-23 Console view

You can then click your job and review the output as shown in Figure 4-24 on page 111.

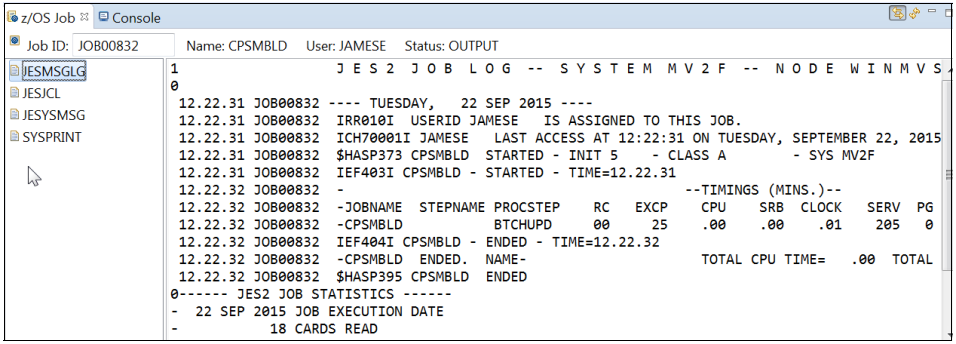


Figure 4-24 z/OS Job output

We have now created CPSM WLM transaction groups. We can add these groups to a WLM definition.

4.2.5 Deploying CPSM transaction groups

We now look at how we can add these transaction groups as a rule to a WLM definition. First, we switch to the SM Administration perspective and create a Workload Specification as shown in Figure 4-25.

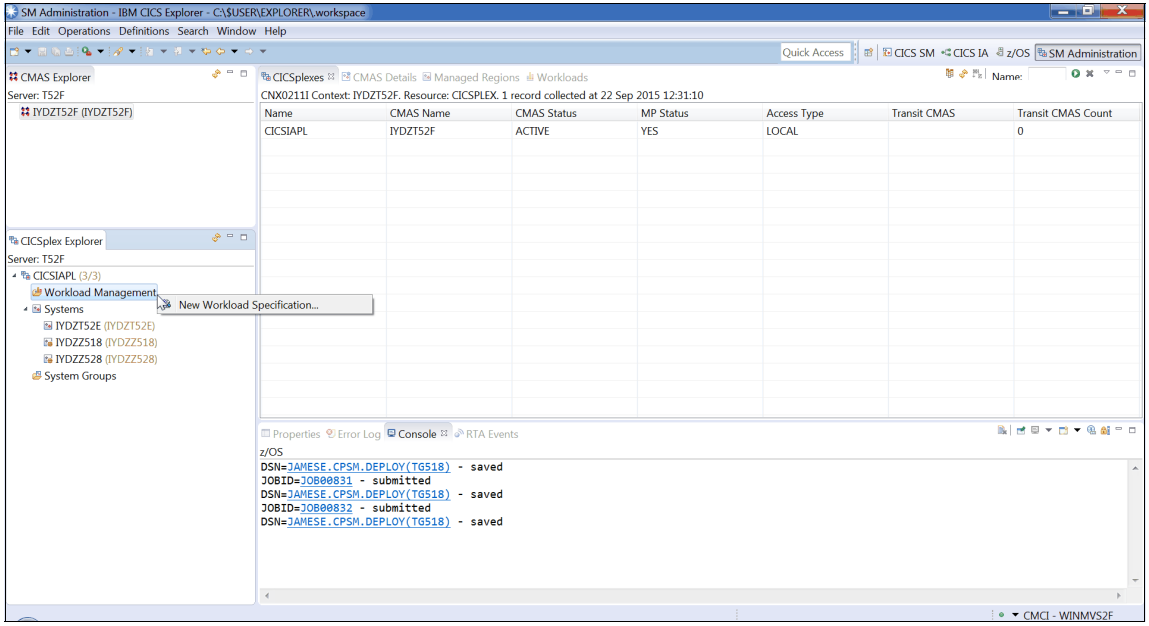


Figure 4-25 Creating a workload specification in the SM Administration perspective

Complete the options in the wizard as shown in Figure 4-26.

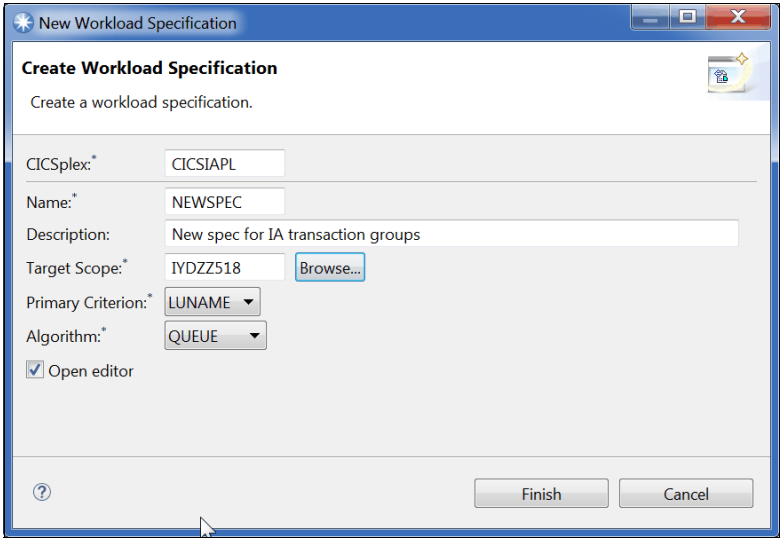


Figure 4-26 New workload management specification

Click **Finish** and the Workload Specification editor opens, as shown in Figure 4-27.

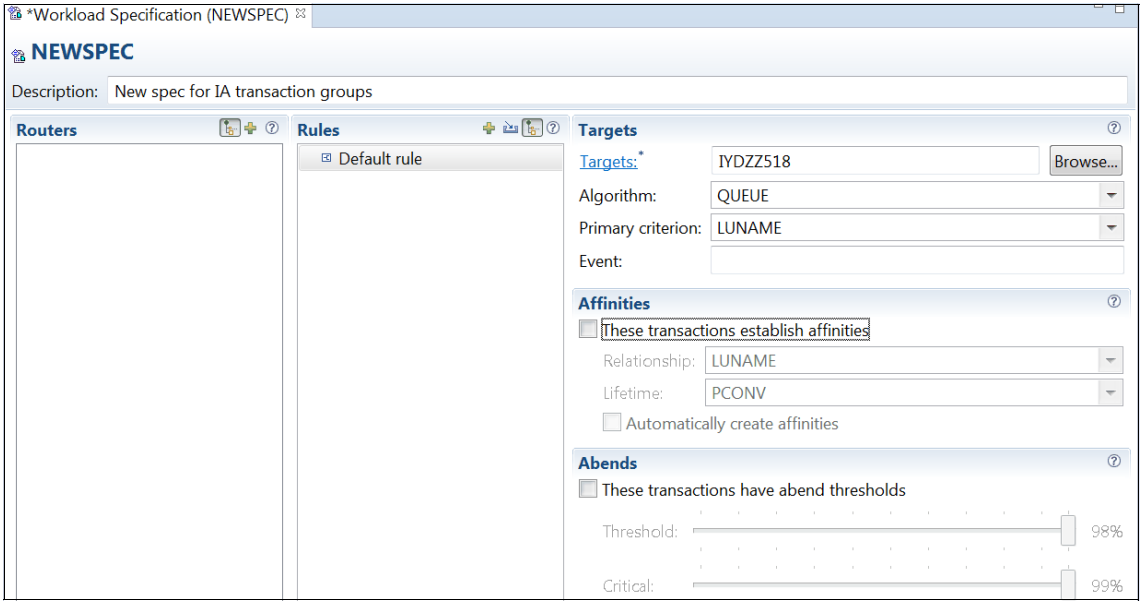


Figure 4-27 Workload Specification editor

You can now add a routing rule as shown in Figure 4-28.

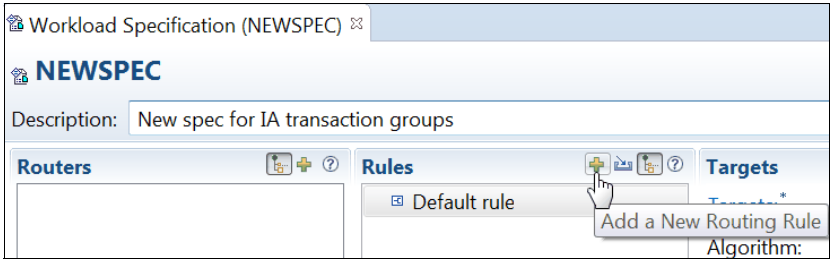


Figure 4-28 Add a routing rule

We can then choose to “Select Transaction Groups” as shown in Figure 4-29.

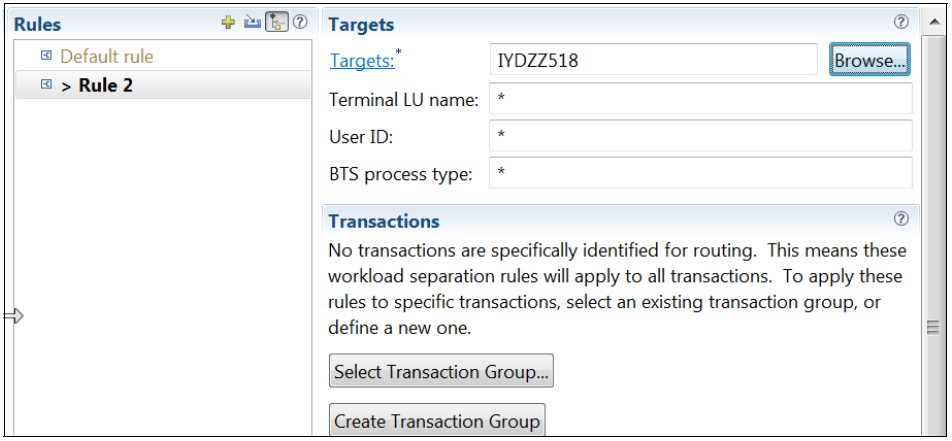


Figure 4-29 Select transaction Group

And then select the group that we just created as shown in Figure 4-30.

We can now see that this group has been added to RULE2 as shown in Figure 4-31 on page 115.

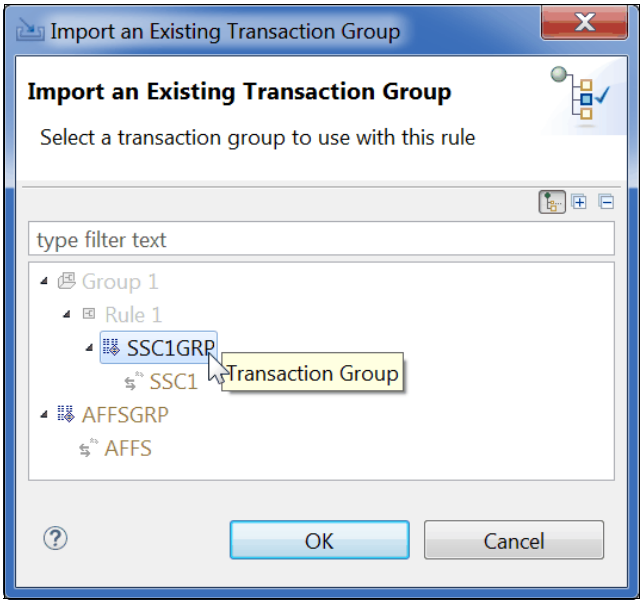


Figure 4-30 Select your group

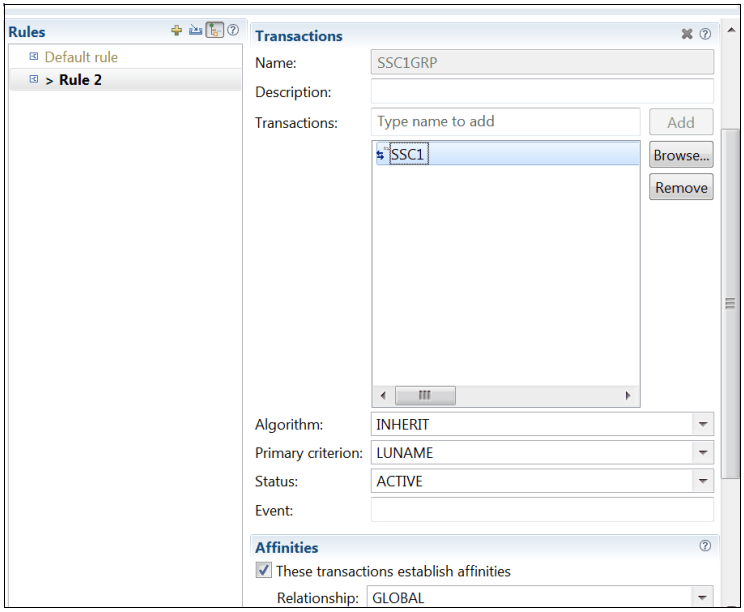


Figure 4-31 New transaction group defined in RULE2

We have successfully deployed a CPSM transaction group created by CICS IA plug-in to a CPSM WLM specification.

5

Using CICS IA data for threadsafe analysis

In this chapter, we look at how CICS Interdependency Analyzer (IA) can be used to assist with ensuring that your applications can execute in a threadsafe manner and take advantage of the performance and throughput benefits of running on open task control blocks (TCBs).

Before using CICS IA to understand the implications of making a program threadsafe, you should use a performance tool such as the CICS Performance Analyzer to identify transactions that have the following specifications:

- Have many TCB swaps
- Are waiting on access to the QR TCB

In this example, we use the transactions used by the GENAPP application:

- SSC1
- SSP1
- SSP2
- SSP3
- SSP4

These transactions are not necessarily good candidates but are used to demonstrate the value of the CICS IA plug-in.

For more information about making your application threadsafe, refer to the popular IBM Redbooks publication: *Threadsafe Considerations for CICS*, SG24-6351. See section “Help from IBM” on page 240.

We look at the following steps:

- 5.1, “What data to collect to enable threadsafe analysis?” on page 118
- 5.2, “Running the load module scanner” on page 124
- 5.3, “Running the threadsafe report in the plug-in” on page 127
- 5.4, “Using the command flow data to analyze TCB swaps” on page 135

5.1 What data to collect to enable threadsafe analysis?

In order to fully analyze and get the best data from running a CICS IA threadsafe report, it is important to collect the correct data. The best data is usually collected from a production region but collecting in a production region introduces performance considerations. Take some time to review how best to collect data in a production region, which is covered in 2.1, “Collecting CICS IA data in a production region” on page 24.

We now review the options that are required for a threadsafe report. We need to collect data for the following information:

- All CICS APIs and SPIs
- Detailed information for programs, files, and connections
- Dynamic COBOL calls
- Remote calls including:
  - DB2
  - IBM MQ
  - IMS/DLI
  - CPSM
- Load module reentrancy information

The CICS IA transaction CINT allows you to control and administer your collections. Figure 5-1 on page 119 shows the main administration menu for the CINT transaction. For more information about the CINT transaction, refer to the *CICS IA Users Guide*, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

```
CIU000        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/07
                          Main Administration Menu                   10:39:38AM


Select one of the following.  Then press Enter.


    1   Operations Menu.
    2   Configure Region Options.
    3   Configure Global Options.
    4   User Administration.









CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82
CIU7000I 5655-Y22 (C) Copyright IBM Corp. 2001,2015
F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Exit
```

Figure 5-1 CINT: Main Administration menu

We use option 2 to “Configure Region options” for a threadsafe collection. Figure 5-2 shows the Region Configuration menu.

```
CIU200        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/07
                         Region Configuration Menu                   10:46:16AM

Type action code then press ENTER.                                   More :
1=Add Region   2=Copy Region   3=Delete Region   4=Options


      CICS      CICS   New       New
 Act  Applid    Sysid  Applid    Sysid  Status       Collecting
      DEFAULTS  DFTS
      ALL       ALL
      IYDZZ518  Z518                   STOPPED
      IYDZZ528  Z528                   STOPPED
      IYDZZ538  Z538                   UNCONNECTED









CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82

F1=Help      F2=           F3=Exit      F4=          F5= Refresh  F6=
F7=Page Up   F8=Page Down  F9=          F10=         F11=         F12=
```

Figure 5-2 CINT: Region Configuration menu

From this menu, we can configure the following options:

- The DEFAULT options for all regions. The defaults are set when the CIUCNTL file is created.
- The REGION-specific options. You can use this to set region-specific options to override the DEFAULT options.
- The ALL options. This is a dummy record that enables you to reset the options for all CICS regions.

We configure the DEFAULTS record to collect dependency data. This is done by selecting line option 4 for “Options” against the DEFAULTS entry.

```
CIU29O        CICS Interdependency Analyzer for z/OS - V5R3M0       2015/09/07
                        Resource Options for                        10:52:43AM
                  CICS Sysid:  DFTS  CICS Applid:  DEFAULTS

Type action code then press ENTER:

    1 = General Options
    2 = Time/Date Options

    Interdependency Options            Affinity Options
    3 = CICS Options for APIs          7 = Affinity Options
    4 = CICS Options for SPIs
    5 = DB2/IMS/MQ/CPSM Options        CICS TS Applications
    6 = Natural Options                8 = Application Data









CICS Sysid: Z528    CICS Applid: IYDZZ528    TermID: TC82

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Exit
```

Figure 5-3 CINT: Region Options for the DEFAULTS record

Figure 5-3 shows the Region Options menu for the DEFAUTS entry. We first look at “General Options” by entering option 1.

Figure 5-4 on page 121 shows the general options for the DEFAULTS record. For a threadsafe dependency collection, we need to collect information about dynamic COBOL calls so set:

```
Dynamic call . . . . . : Y    (Y=Yes, N=No)
```

We also need to ensure that we are collecting for dependency data only so set:

```
Data to Collect  . . . . . : I
```

```
CIU260        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/07
                          General Options for                        10:57:22AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS
Modify the options and press Enter to update, or PF12 to Cancel.
Control options
  Data to Collect  . . . . . . . : I    (A=Affinity, I=Interdependency, B=Both)
  Perform periodic saves . . . : N    (Y=Yes, N=No)
  Trigger for CINB start . . . : 1    (2 to 9999 thousand record updates)
  Restore data on start  . . . : N    (Y=Yes, N=No)
  Maintain usage counts  . . . : Y    (Y=Yes, N=No)
  Size of dataspace  . . . . . : 16   (10 to 2000 Mbytes)
  Transid prefix (optional). . :      (1 to 4 characters)
  Program exclude list . . . . : CIUXPROG (1 to 8 characters)
  Transaction exclude list . . : CIUXTRAN (1 to 8 characters)
  Command exclude list . . . . : CIUXCOMM (1 to 8 characters)
  Resource compression list. . : CIUXRCOM (1 to 8 characters)
  Dump HLQ . . . . . . . . . . : DUMP     (1 to 8 characters)
  Dynamic call . . . . . . . . : Y    (Y=Yes, N=No)
  Trigger for Task collection. : 1    (1 to 9999)
  Collect Long Running Tasks . : N    (Y=Yes, N=No)
CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82

F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 5-4 CINT: General Options for the DEFAULTS entry

Next, we need to set the CICS APIs and the SPIs we collect. From the Regions Options menu in Figure 5-3 on page 120, select option 3 for CICS API options.

Figure 5-5 on page 122 shows the CICS “APIs” options panel. For a threadsafe collection, we need to collect DETAILED data for Programs and Files. So we need to set:

```
*Programs . . . D *Files. . . . . D
```

Now from the “Regions Options” menu in Figure 5-3 on page 120, select option 4 for CICS SPI options.

Figure 5-6 on page 122 shows the CICS “SPIs” options panel. For a threadsafe collection, we need to collect DETAILED data for connections. So we need to set:

```
*Connections  . D
```

This captures data about your inter-region connections and reports them as IPIC or MRO. We need this information to determine whether your remote CICS calls are threadsafe.

```
CIU240        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/07
                        CICS Resources Options for                   11:10:09AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.

Detect command types: Y=Yes, N=No
                      D=Yes+Detail ( Only for API types marked with * )
APIs
*Programs . . . D *Files. . . . . D *Transactions . D  Task Control . Y
 Presentation . Y *TS Queues  . . D *TD Queues  . . D  Journals . . . Y
 DTP  . . . . . Y  Counters . . . Y  FEPI . . . . . Y *WEB Services . D
*Exits  . . . . D  Others . . . . Y *EVENTS . . . . D  ATOMServices . Y
 XMLtransform . Y  WSAddressing . Y









CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 5-5 CINT: CICS API options

```
CIU245        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/07
                        CICS Resources Options for                   11:13:50AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.

Detect command types: Y=Yes, N=No
                      D=Yes+Detail ( Only for SPI types marked with * )
SPIs (Create/Inquire/Set/Discard/Perform)
 Programs . . . Y  Files  . . . . Y  Transactions . Y  Temp Storage . Y
 Transient Data Y  DB2  . . . . . Y  DJAR . . . . . Y  BRFacility . . Y
 Corbaserver  . Y  TCPIPService . Y  FEPI . . . . . Y  Journals . . . Y
 Library  . . . Y *Connections  . D  BTS proc . . . Y  Bundles  . . . Y
 ATOMServices . Y  CSD. . . . . Y  XMLTransform . Y  MQCONN . . . . Y
 JVMServices .  Y  Terminals. . . Y  CICS System  . Y  Tasks  . . . . Y
 DUMPs . . . .  Y  VTAM Conn . .  Y  Statistics . . Y  Tracing . . .  Y
 SHUTDOWN . . . Y



CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 5-6 CINT: CICS SPI options

Last, we need to check the remote call options. From the Regions Options menu in Figure 5-3 on page 120, select option 5 for the DB2/IMS/MQ/CPSM Options.

Figure 5-7 shows the remote call options. If your environment uses any of these options, set the option to Y.

```
CIU25O        CICS Interdependency Analyzer for z/OS - V5R3MO        2015/09/07
               DB2/MQ/IMS/CPSM       Resource Options for            11:21:25AM

          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.
Detect command types: Y=Yes, N=No

DB2 Options
  Collect DB2 Resources . . . . . : Y
  Collect resource name . . . . . : Y (Y/N) (Access the SYSIBM.SYSPACKSTMT
                                      and the SYSIBM.SYSSTMT tables)
MQ Options
  Collect MQ Resources  . . . . . : Y

IMS Options
  Collect IMS Resources . . . . . : Y

CPSM Options
  Collect CPSM Resources  . . . . : Y
CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC82

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F1O=         F11=         F12=Cancel
```

Figure 5-7 CINT: Remote call options

We should have now set all the options required for a threadsafe collection.

Now you can start the collector, capture the data, and load the dependency data into the database. To load the data, you can run sample job CIUUPDB. Before submitting the job, change the collection ID to something meaningful that will help you identify your threadsafe collection. In our example, we use a collection ID of CICS510. In the ISPF editor for CIUUPDB, issue the following command and then run the job:

```
C _collid_ CICS51O all
```

Next, we look at how we run a threadsafe report in the CICS IA plug-in from the data we captured.

5.2 Running the load module scanner

Before you can make a program threadsafe, it must be linked with the attribute “reentrant”. CICS IA provides a load module scanner, which captures this information. In this section, we look at how to run a scanner and load the data into DB2 tables:

- CIU_SCAN_SUMMARY
- CIU_SCAN_DETAIL

The data store in these tables is used when we run a threadsafe report to obtain the reentrancy link attribute.

First, we must run a summary report against the load module data set. The sample JCL to run a summary report, which loads the data into DB2 can be found in SCIUSAMP(CIUJCLTS). You need to update the input load module data set name and the output data set name that holds the list of load modules that is used as input when running the DETAILED report. See Figure 5-8; we scan the load module data set “CICSIAD.GENAPP.V51.LOAD”.

```
//*   _scan_                                                      *
//*   CICS LOAD DATASET TO BE SCANNED                             *
//*                                                               *
//*   _ciudet_                                                    *
//*   Output dataset created by SCANNER SUMMARY JOB               *
//*                                                               *
......
//INPUT    DD DSN=CICSIAD.GENAPP.V51.LOAD,
//            DISP=SHR
......
//INTMOD   DD DSN=CICSIAD.GENAPP.DETAIL,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000),SPACE=(CYL,(1,1))
```

Figure 5-8 Input for SUMMARY report

This job loads summary data into the DB2 table CIU_SCAN_SUMMARY and also produces a report as shown in Figure 5-9 on page 125.

```
CICS INTERDEPENDENCY ANALYZER  Version 5.3.0
LOAD MODULE SCANNER -  SUMMARY LISTING OF CICSIAD.GENAPP.V51.LOAD

Module      Module    Module       Re-       Language   Possible
Name        Length    Language     entrant   Version    Affinities   Dependencies   MVS
--------   --------   ----------   -------   --------   ----------   ------------
LGACDB01   00002540   COBOL II     Y         LE              0            9          0
LGACDB02   00001D60   COBOL II     Y         LE              0            6          0
LGACUS01   00001BC8   COBOL II     Y         LE              0            7          0
LGACVS01   00001BC8   COBOL II     Y         LE              0            9          0
LGAPDB01   00003868   COBOL II     Y         LE              0           12          0
LGAPOL01   00001B98   COBOL II     Y         LE              0            8          0
LGAPVS01   00001AA8   COBOL II     Y         LE              0            6          0
LGASTAT1   00001AA0   COBOL II     Y         LE              2            7          0
LGDPDB01   00001D30   COBOL II     Y         LE              0            7          0
LGDPOL01   00001B88   COBOL II     Y         LE              0            7          0
LGDPVS01   000019F8   COBOL II     Y         LE              0            6          0
LGICDB01   00001F70   COBOL II     Y         LE              0            7          0
LGICUS01   00001D58   COBOL II     Y         LE              2           10          0
LGICVS01   00001E50   COBOL II     Y         LE              7           10          0
LGIPDB01   00005E48   COBOL II     Y         LE              0            9          0
LGIPOL01   00001A88   COBOL II     Y         LE              0            7          0
LGIPVS01   00001870   COBOL II     Y         LE              0            3          0
LGSETUP    00003CA8   COBOL II     Y         LE              7           83          0
LGSTSQ     00001888   COBOL II     Y         LE              1            4          0
LGTESTC1   00002B10   COBOL II     Y         LE              8           26          0
LGTESTP1   00002898   COBOL II     Y         LE              0           22          0
LGTESTP2   000026E0   COBOL II     Y         LE              0           21          0
LGTESTP3   000026C8   COBOL II     Y         LE              0           21          0
LGTESTP4   00002868   COBOL II     Y         LE              0           16          0
LGUCDB01   00001E50   COBOL II     Y         LE              0            7          0
LGUCUS01   00001AA8   COBOL II     Y         LE              0            7          0
LGUCVS01   00001B88   COBOL II     Y         LE              0            9          0
LGUPDB01   000031B8   COBOL II     Y         LE              0            8          0
LGUPOL01   00001C60   COBOL II     Y         LE              0            7          0
LGUPVS01   00001C80   COBOL II     Y         LE              0            9          0
LGWEBST5   00004D68   COBOL II     Y         LE             10           49          0
SSMAP      000028B0                                         0            0          0
CICS INTERDEPENDENCY ANALYZER  Version 5.3.0
LOAD MODULE SCANNER -  SUMMARY LISTING OF CICSIAD.GENAPP.V51.LOAD


                  LOAD LIBRARY STATISTICS
================================================================
Total modules in library                          =       32
Total modules scanned                             =       32
Total CICS modules/tables (not scanned)             =       0
Total modules in error (not scanned)                =       0
Total modules containing possible MVS POSTs           =       0
Total modules containing possible Dependency commands =      31
Total modules containing possible Affinity commands   =       7
  Total ASSEMBLER modules                         =       0
  Total C/370 modules                             =       0
  Total COBOL modules                             =       0
  Total COBOL II modules                          =      31
  Total PL/I modules                              =       0
Total number of possible Dependency commands        =     419
Total number of possible Affinity commands            =       37
```

Figure 5-9 Sample summary report

We now need to run the DETAIL report to obtain the linkage attributes for the load modules in load library CICSIAD.GENAPP.V51.LOAD. The sample JCL to run a DETAILED report, which loads the data into DB2, can be found in SCIUSAMP(CIUJCLTD). Again, you need to update the input load module data set name and the detail input data set name that holds the list of load modules that was created by running the SUMMARY report. See *5.3, "Running the threadsafe report in the plug-in" on page 127*; we scan the load module data set "CICSIAD.GENAPP.V51.LOAD".

```
//*   _scan_                                                          *
//*   CICS LOAD DATASET TO BE SCANNED                                 *
//*                                                                   *
//*   _ciudet_                                                        *
//*   Input dataset created from a SCANNER SUMMARY JOB                *
//*                                                                   *
......
//INPUT    DD DSN=CICSIAD.GENAPP.V51.LOAD,
//            DISP=SHR
.....
//DETAIL   DD DSN=CICSIAD.GENAPP.DETAIL,
//            DISP=(OLD,DELETE)
```

Figure 5-10 Input for DETAIL report

This job loads the DETAIL data into the DB2 table CIU_SCAN_DETAIL and also produces a report as shown in Figure 5-11 on page 127.

```
LOAD MODULE SCANNER - DETAILED LISTING OF CICSIAD.GENAPP.V51.LOAD

Module Name - LGACDB01  /  Load Module Length - 00002540  /  Module Entry Point - 00000020
Offset    Storage Content (HEX)                                EDF DEBUG  Possible Command         Type
Depcy   Affinity
--------  ---------------------------------------------------  ---------  ------------------------ ----------------
------  --------
00000266  4A048002078002080000000000                             00057   FORMATTIME               TIME
Yes
0000027B  4A028000078000000000000000                             00057   ASKTIME ABSTIME          TIME
Yes
00000290  2006D000270000010200000101                             00047   GET                      COUNTER
Yes
000002CC  0E02E0000700000100                                     00059   LINK                     PROGRAM
Yes
000002DD  0E02E0000700000100                                     00059   LINK                     PROGRAM
Yes
000002EE  0E02E0000700000100                                     00058   LINK                     PROGRAM
Yes
00000343  0E02E0000700000100                                     00045   LINK                     PROGRAM
Yes
00000354  0E02E0000700000100                                     00044   LINK                     PROGRAM
Yes
00000376  0E0C80000700006080                                     00042   ABEND                    PROGRAM
Yes
Total possible Affinity commands   =        0
Total possible Dependency commands =        9
Total possible MVS POSTs           =          0
```

Figure 5-11 Sample DETAILED report

5.3 Running the threadsafe report in the plug-in

After we collected and loaded threadsafe data into the DB2 database, we can then run a threadsafe report in batch by using sample job CIUJTSQ2 or in the CICS IA plug-in. In this book, we run the report in the plug-in.

First, we reduce the amount of data we query in the plug-in by setting a scope to the collection ID that we used in the CIUUPDB load job, in our case CICS510. In the Collection IDs view, right-click your collection ID, CICS510, and select “Set as current scope” as shown Figure 5-12.

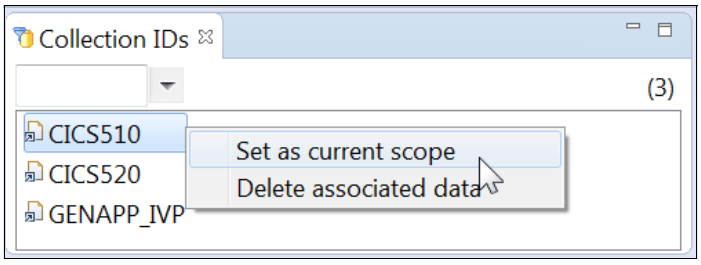


Figure 5-12 Set current scope

You can run a threadsafe report against a CICS region, a transaction, or a program. In this book, we run a region Thursdays report. From the `Regions` folder in the IA Navigation view and against your CICS region, select **Report → Threadsafe Report**, as shown in Figure 5-13.

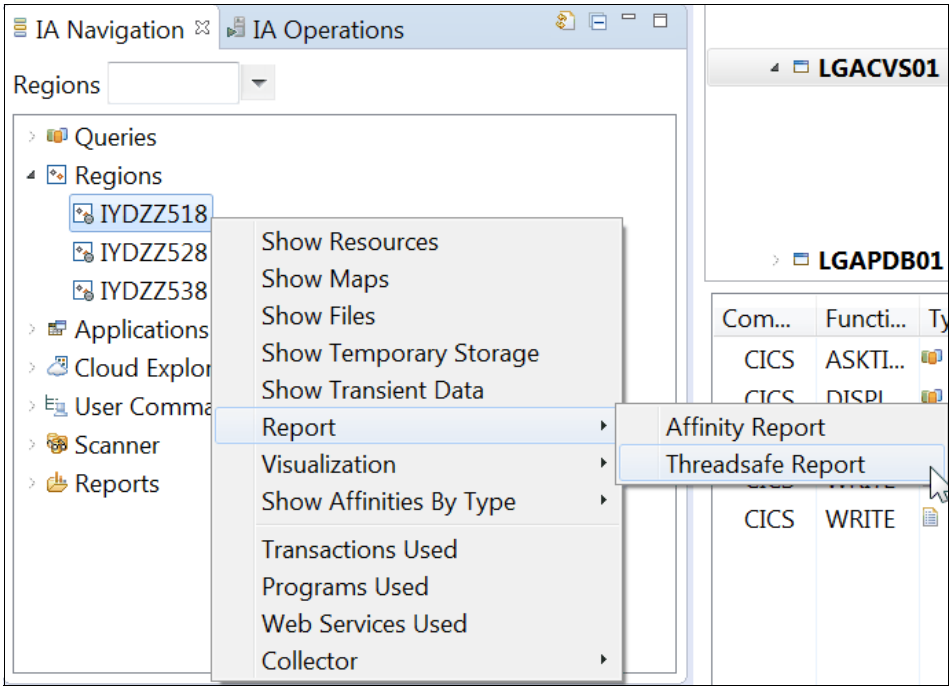


Figure 5-13 Running a threadsafe report

This action opens the Threadsafe Report view as shown in Figure 5-14 on page 129. This report is broken into two parts. The top part shows the list of programs and a corresponding summary for each program. To see the summary, select the program and the following data is shown.

Total CICS calls

This value is the total number of CICS calls.

Threadsafe calls

This value is the total number of threadsafe commands, that is, commands that can run an open TCB and do not cause a swap back to the QR TCB.

Non-threadsafe calls

This value is the total number of CICS commands that are not threadsafe in that they need to run on the QR TCB and therefore can cause a TCB swap from an open TCB back to the QR TCB.

Total number of indeterminate calls

This value is the number of CICS calls where we do not have sufficient information to decide whether the call is threadsafe or not.

Total number of DB2 calls

This value is the total number of DB2 calls. DB2 calls moved to run on the open TCB in CICS TS V2.2.

Total number of IBM MQ calls

This value is the total number of IBM MQ calls. IBM MQ calls moved to run on the open TCB in CICS TS V3.2.

Total number of IMS calls

This value is the total number of IMS calls. IMS/DLI calls moved to run on the open TCB in CICS TS V4.2 when connected to the required level of IMS that supports this feature.

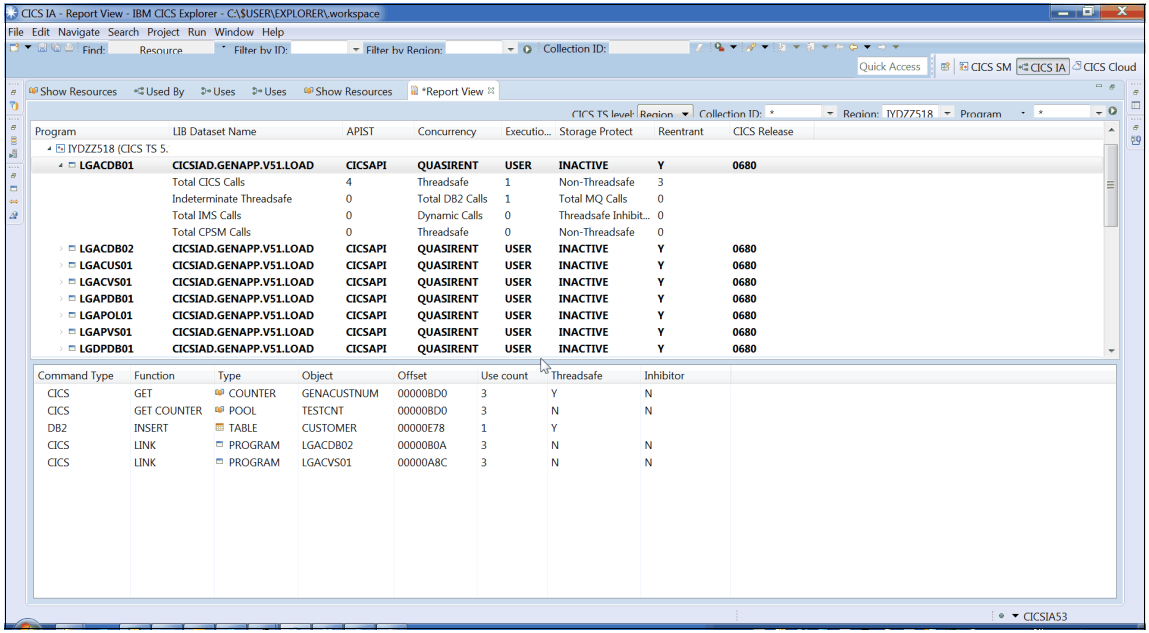


Figure 5-14 Threadsafe Report view

Dynamic calls

This value is the total number of dynamic COBOL calls that have been identified. You need to know if a program is dynamically called because a dynamically called program takes its resource definition attributes from the calling program. For example:

- Program PROGA is defined with CONCURRENCY QUASIRENT
- Program PROGB is defined with CONCURRENCY THREADSAFE
- Program PROGA dynamically calls program PROGB

You might think that program PROGB is now running as threadsafe but in fact it is still running as *quasi-reentrant*. This fact can be even more critical if PROGB calls PROGA. You might have decided that PROGB can be threadsafe but PROGA must remain as quasi-reentrant because it has commands that are threadsafe inhibitors and could lead to data integrity issues if it runs on an open TCB.

Threadsafe inhibitor calls

This value is the total number of CICS calls that could lead to data integrity issues. Following are the commands:

- ADDRESS CWA
- GETMAIN SHARED
- ENABLE EXIT PROGRAM
- LOAD HOLD

All of these commands are used to address areas of storage associated with the program. Before making a program that contains any of the commands threadsafe further investigation is needed to understand if the area of storage addresses is read only or can be updated. Programs using shared data must be serialized. This serialization can be done by leaving the program as quasi-reentrant so it goes back to the QR TCB. Or if you want to run in an open environment, you must use one of the following serialization techniques:

- CICS API enqueue/dequeue
- CICS SPI enqueue/dequeue
- Compare and swap

Total number of CPSM calls

- Threadsafe
- Non-Threadsafe

In CICS IA V5.2, we added the collection of CPSM commands. Currently, all CPSM commands are not threadsafe and run on the QR TCB.

For each program we also show:

- Library data set name
- API attribute
- Concurrency attribute
- Program storage execution key
- Region storage protection key
- Linkage reentrancy indicator
- CICS release level

These facts are all good to know when analyzing whether to make your program threadsafe.

The lower part of the Threadsafe Report view as shown in Figure 5-14 on page 129 shows us the actual commands issued by the program selected in the top part. We show the following commands:

Command type	CICS, DB2, MQ, IMS, or CPSM
Function	The actual command, for example, LINK
Type	The resource type, for example, PROGRAM
Offset	Offset of the command within the program
Use Count	The number of times the command has been invoked
Threadsafe	For a CICS command, is it threadsafe
Inhibitor	For a CICS command, is it a possible inhibitor

Selecting the threadsafe report input parameters

The threadsafe report view allows you to change the input parameters and rerun your threadsafe report from the toolbar found in the top of the view as shown in Figure 5-15.

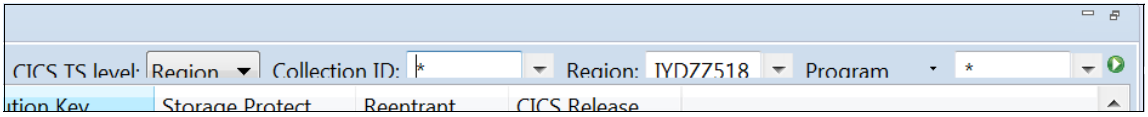


Figure 5-15 Threadsafe input parameters

You can change the following input options and then select the green arrow icon to rerun the report:

- Collection ID
- Region
- Program
- CICS TS Level

The CICS TS Level input parameter can be set to the CICS release level in which the data was collected or can be set to a specific release level as shown in Figure 5-16. This feature can be very valuable if you collected data on your existing CICS release say V4.1, but want to see what the report would look like if you went to CICS V5.2.

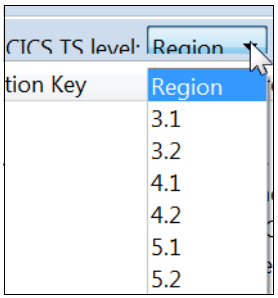


Figure 5-16 Choose your CICS level

5.3.1 Saving and printing threadsafe reports

In this section, we save our threadsafe report and then retrieve it for printing.

First, ensure that the threadsafe report view is active then select the option **File → Safe** from the upper-left corner or enter CTRL+S. The Save report window opens. Choose the threadsafe folder and enter an appropriate name as shown in Figure 5-17.

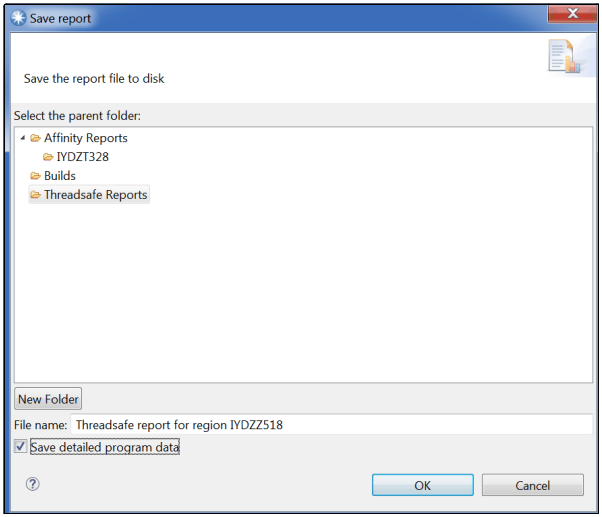


Figure 5-17 Saving a threadsafe report

Select **OK** and your report is saved.

Note: To save detailed information, you must first retrieve the detailed information by selecting the program and tick the “Save detailed program data” box in the Save report view.

All reports are stored in the `Reports` folder in the IA Navigation view. To retrieve your report, right-click the report and select Open Report as shown in Figure 5-18.

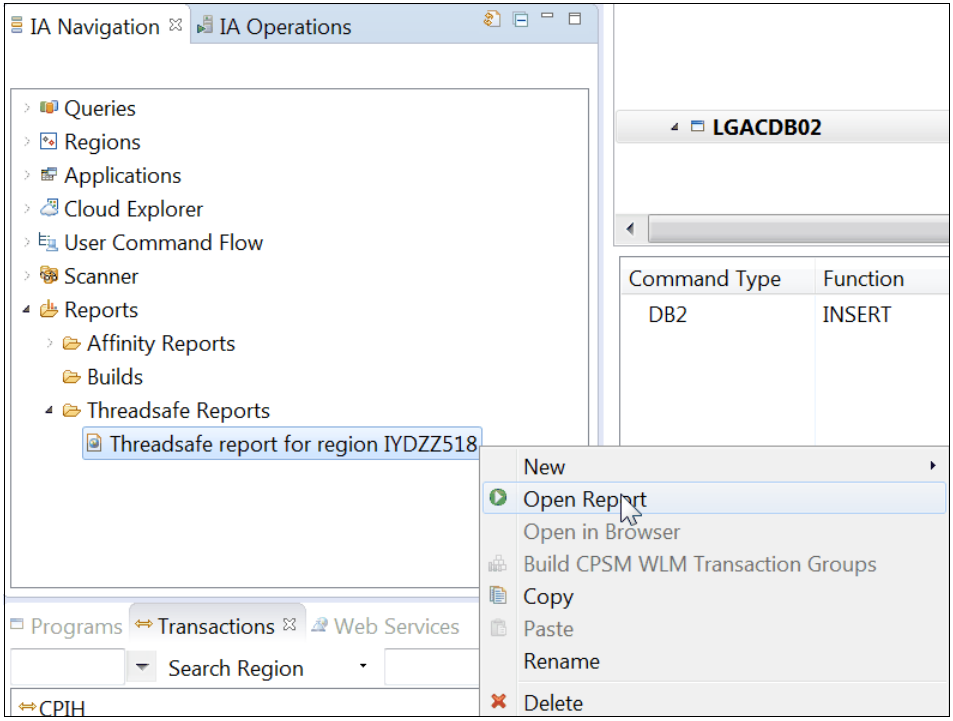


Figure 5-18 Opening a threadsafe report

The report is then displayed in a simple HTML format as shown in Figure 5-19.

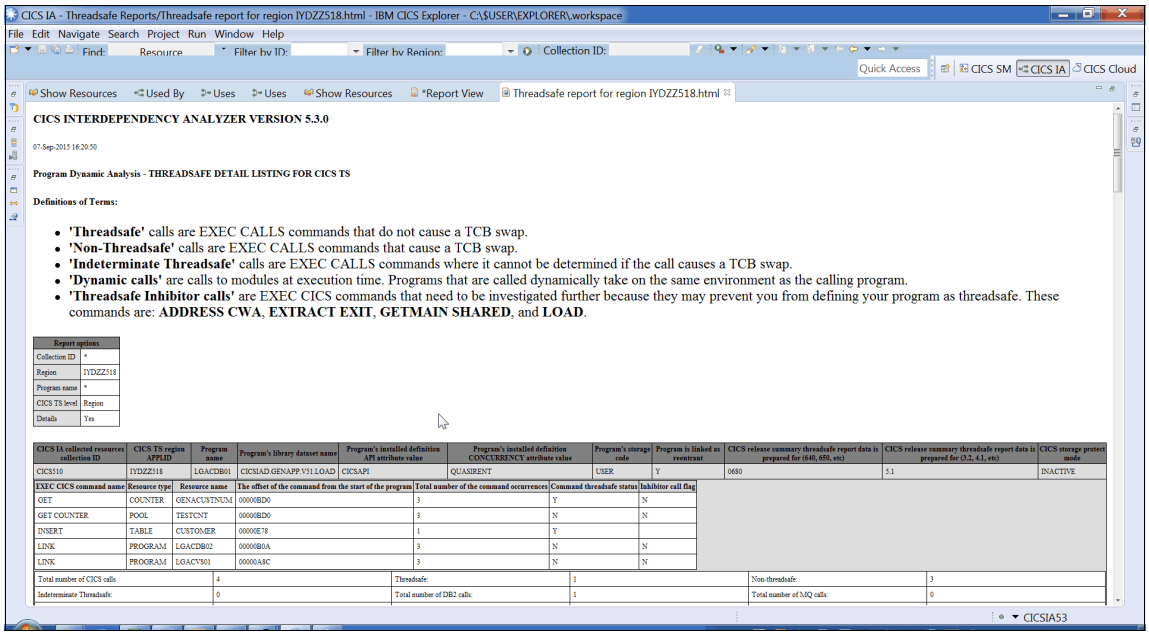


Figure 5-19 HTML threadsafe report

To print the report, right-click anywhere within the report and select “Print” Preview” or “Print,” as shown in Figure 5-20.

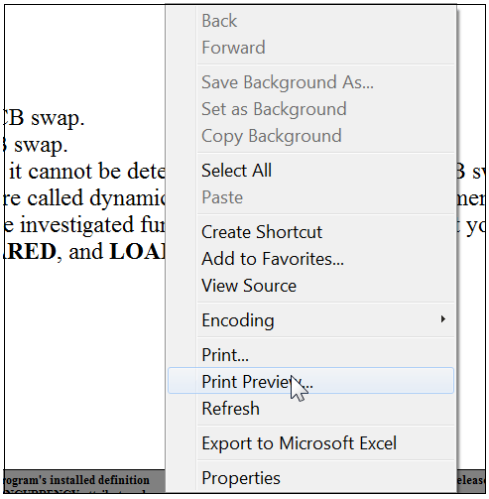


Figure 5-20 Select “Print Preview”

5.4 Using the command flow data to analyze TCB swaps

As well as using the CICS IA threadsafe report to assist with threadsafe analysis, you can also use the Command Flow feature. The Command Flow feature was originally added for exactly this function and can assist you in identifying where the actual TCB switches occur.

Information about how to configure, load, and use the Command Flow data can be found in Chapter 8, “Command Flow feature” on page 197.

In this section, we focus on the Command Flow features that assist with threadsafe analysis. We use the Command Flow data captured for transaction SSC1 as described in 8.1.3, “Using the IA plug-in to collect Command Flow data” on page 204.

5.4.1 Identifying TCB switches

Right-click the required task and select **Show Execution** as shown in Figure 5-21.

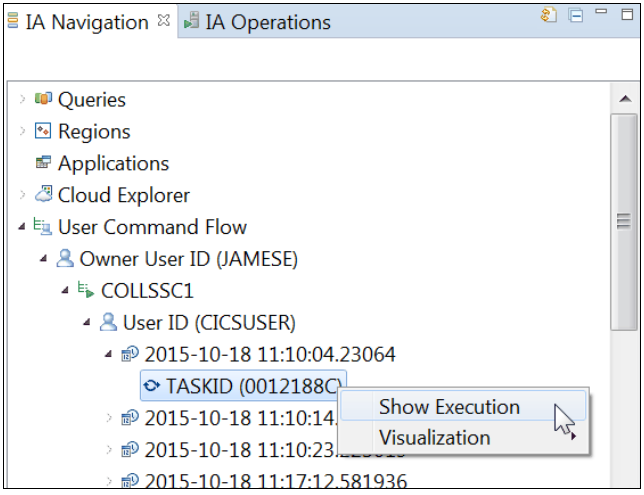


Figure 5-21 Show Command Flow execution

This action opens the Command Flow view as shown in Figure 5-22.

This view is broken down into three parts:

- TCB Modes Used: The command execution count by TCB mode
- TCB Mode Switches: The command count causing a TCB mode switch
- The execution tree: The command execution tree

In this section, we look at the first two parts. The execution tree is described in detail in 8.3.1, “The Command Flow Execution view” on page 213.

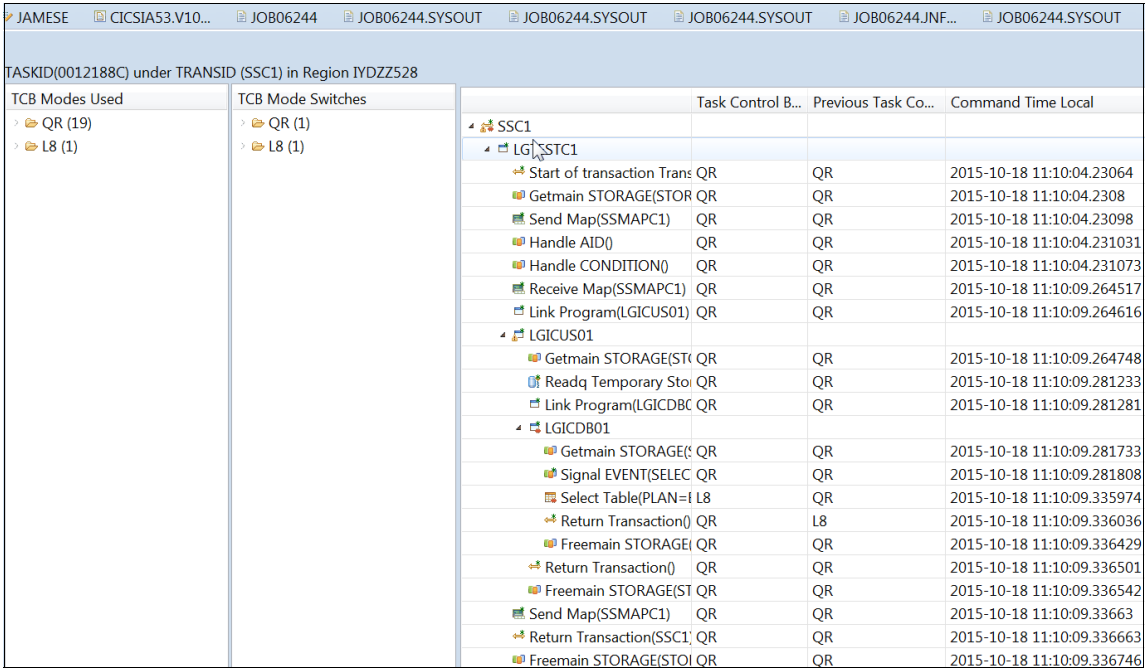


Figure 5-22 Command Flow execution

TCB modes used

This section lists all the commands used under each TCB mode, as shown in Figure 5-23.

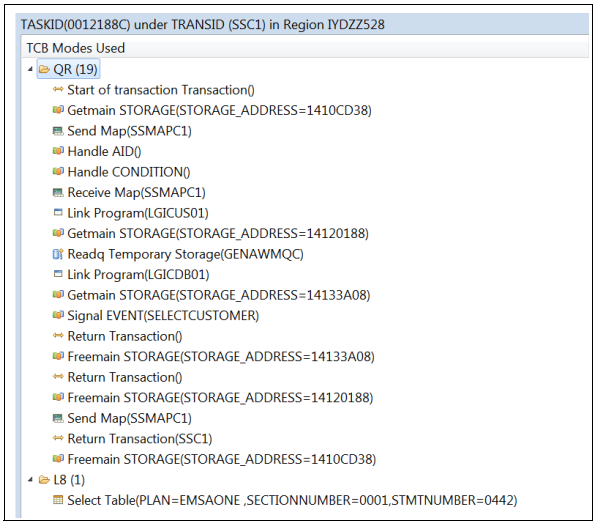


Figure 5-23 Commands by TCB mode

TCB mode switches

This section lists the commands that cause a TCB switch, as shown in Figure 5-24.

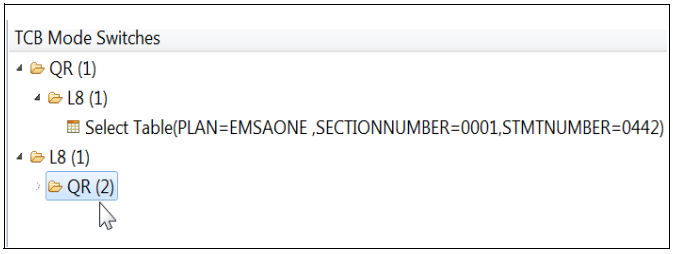


Figure 5-24 TCB mode switches

You can see that one DB2 SELECT command causes a switch from the QR TCB to the L8 TCB.

If you select the command, it takes you to that command in the execution part of the view, as shown in Figure 5-25.

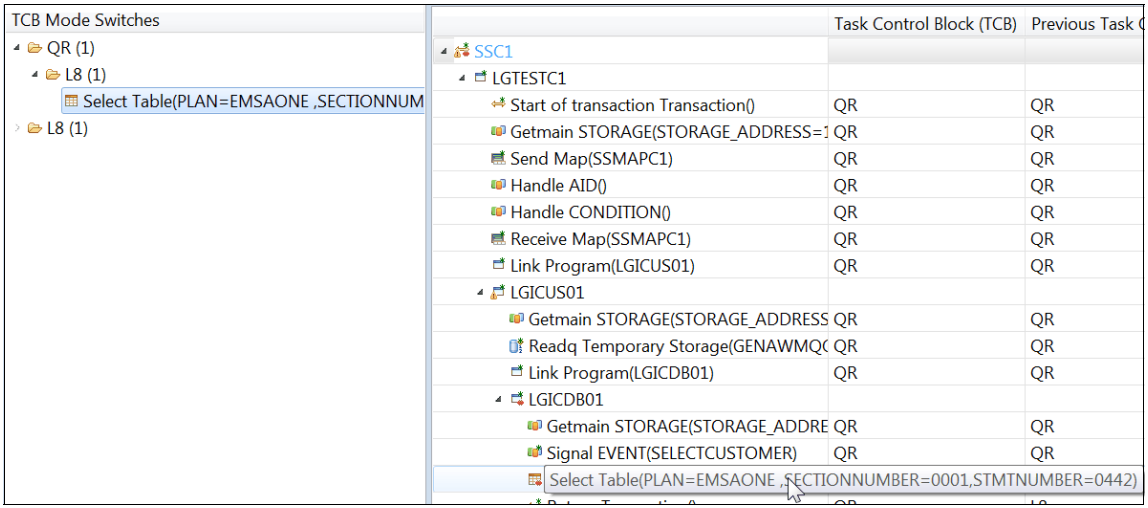


Figure 5-25 Select the DB2 command

If you look carefully, you also see a red diamond icon on the command as shown in Figure 5-26. This indicates that this command causes a TCB swap.

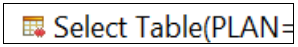


Figure 5-26 The red diamond ICON

For threadsafe analysis, you might want to modify the view to add the columns shown in Figure 5-27.

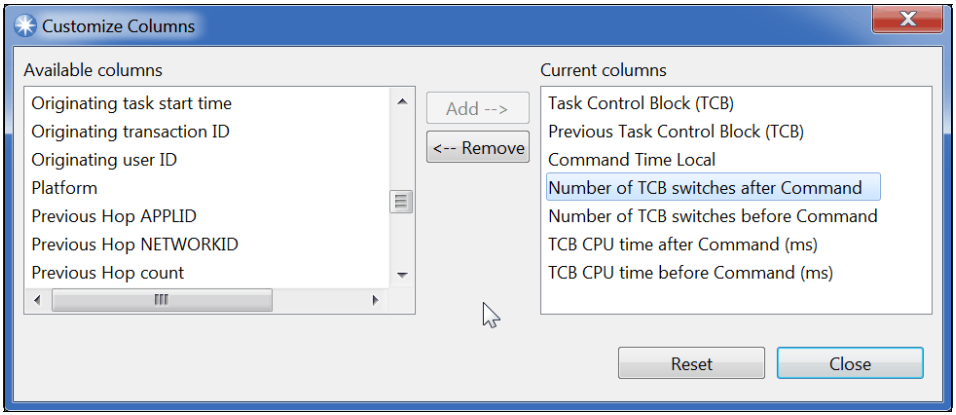


Figure 5-27 Add more columns

For more information about customizing columns, go to “Customizing columns” on page 215.

6

Application onboarding

In this chapter, we show how CICS Interdependency Analyzer (IA) can assist in identifying application entry points and application dependencies to enable you to cloud-enable your CICS applications.

This chapter should be read and used with IBM Redbooks publication *Cloud Enabling IBM CICS*, SG24-8114. See section “Help from IBM” on page 240.

We show how CICS IA can assist in the following approaches:

- 6.1, “CICS cloud capabilities” on page 143
- 6.2, “Identifying entry points and dependencies: One-phase approach” on page 144
- 6.3, “Identifying entry points and dependencies: Two-phase approach” on page 161
- 6.4, “View application data” on page 174

In this publication, we will not create and deploy platforms and applications. That is covered in the “Cloud Enabling CICS” Redbooks publication.

Note: The new Application Entry points and Dependency wizards used in the creation of this book were still in BETA development. They might not exactly look the same as the ones in the final product.

Before that, we give a brief overview of the CICS cloud capabilities. The following extract is copied from the “Cloud Enabling IBM CICS” Redbooks publication mentioned above.

6.1 CICS cloud capabilities

Cloud computing, although a relatively new concept, offers many values that existing CICS Transaction Server (TS) users claim they already enjoy, so why introduce new cloud style concepts into CICS TS today?

Well, cloud is more than just a technology. Cloud offers a conceptual shift in how a business delivers services. It promises increased operational efficiency over the management and operation of these services, and increased agility when developing and deploying them. By introducing cloud capabilities, CICS TS allows both new and existing customers to gain the benefits offered by cloud computing, while maintaining the solid foundation CICS TS offers.

Specifically, CICS TS provides the following cloud capabilities and benefits.

Platforms as first-class entities

First-class platforms enable the creation of agile service delivery runtimes. CICS TS regions can be grouped as platforms for rapid application deployments, decoupling applications from the underlying topology, which increases flexibility. When regions within a platform are started, applications are deployed to them, without any further interaction from a system administrator. In turn, reliability is increased through automatic resource validation, provisioning, and de-provisioning. Platforms can be managed dynamically by applying policies during runtime.

Applications as first-class entities

First-class applications enable the creation of agile services from new or existing assets. Disparate application resources can be combined and managed as a single entity that can be versioned and rapidly moved through the development, test, and production lifecycle. Using applications improves dependency management, and entire applications can be measured for resource usage and internal billing. Applications can be managed dynamically by applying policies during runtime.

Policy-based operations

Automated control over critical system resources can now be managed by using policies. Task thresholds can now be set for data access requests, storage usage, program loops, and processor time used. Policy breaches can be managed by issuing messages, abending tasks, or emitting events that can trigger further actions. Policies can be applied dynamically during runtime operation.

These three capabilities, when combined with the existing features of CICS Transaction Server, provide the building blocks to enable you to transform your existing CICS TS topologies and applications into cloud-style platforms and services.

6.2 Identifying entry points and dependencies: One-phase approach

In this section, we use a technique to capture *dummy* operational data for a CICS transaction. We then use the CICS IA plug-in wizards to identify both the entry points and dependencies for this operation. The wizards will also be used to create a CICS Bundle that can then be included when you define your CICS Application. Creating a CICS Application is covered in chapter 4, “Create CICS Application” of the “*Cloud Enabling IBM CICS*” Redbooks publication.

In this section, we perform the following steps:

- Assign an operation value to the initial program for transaction SSC1
- Collect CICS IA Application data for transaction SSC1
- Identify and create an entry point CICS Bundle
- Identify and create a dependency Bundle

6.2.1 Assign an operation value to the initial program for transaction SSC1

In CICS TS V5 onwards, you can edit a program entry and assign an operation value to the Operation field associated with it. You can perform this task by using the CEMT transaction, the CICS Explorer resource editor, or from within the CICS IA plug-in perspective. We do it from the CICS IA plug-in.

Restriction: At the time of publication, the ability to use CICS IA to capture data that is based on an assigned operation value is only possible in CICS TS V5.1. Work is in progress to make this capability available in later releases.

First, we need to identify the initial program for transaction SSC1. Right-click the transaction and select **Uses Resources → Specific Region** as shown in Figure 6-1.

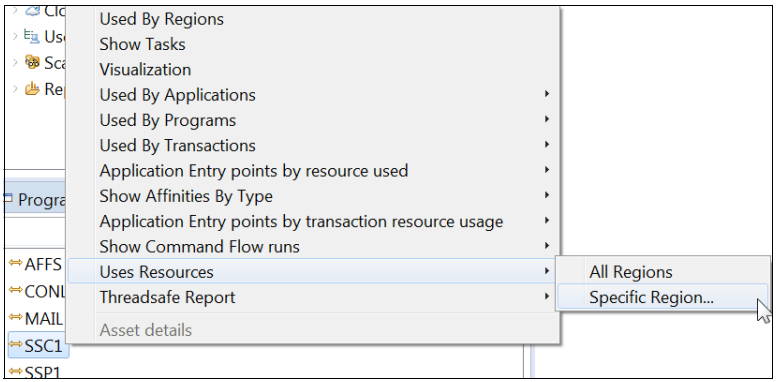


Figure 6-1 Uses resources for SSC1

We select region IYDZZ528. This action opens the Uses view as shown in Figure 6-2. For more information about the Uses view, refer to 3.1.2, “The Uses view” on page 67.

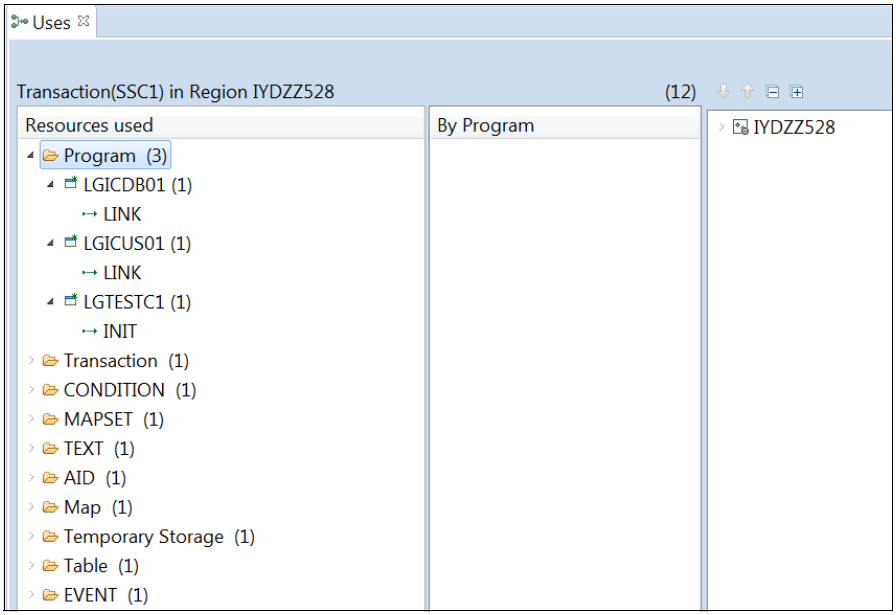


Figure 6-2 Uses view

By fully expanding the expanding “Program entry” in the “Resources used” window, you can see that the initial program for SSC1 is LGTESTC1.

We can now open the Explorer editor for this program and set an operational value. Right-click the program and select **Open CICS SM Editor** as shown in Figure 6-3.

Note: For the editor to open, you must have an active CMCI connection to your CPSM environment and the program must be active.

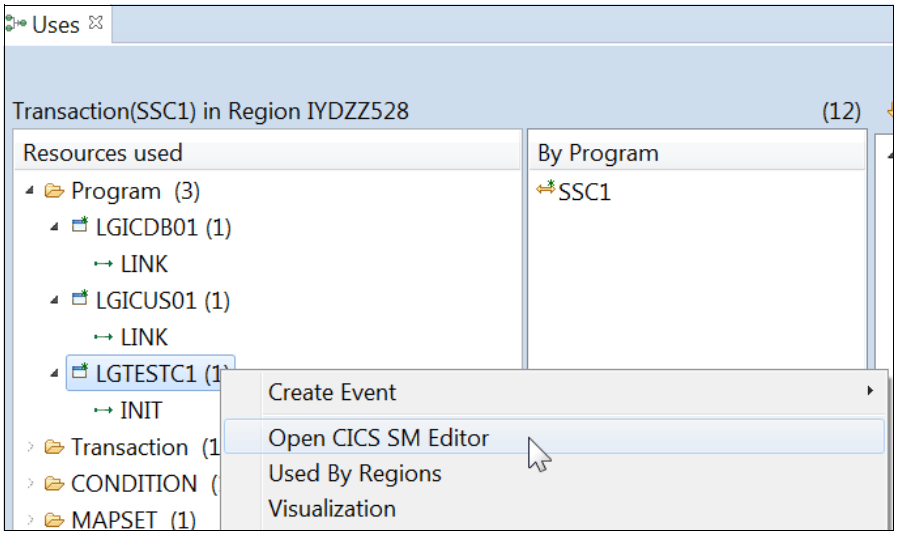


Figure 6-3 Open CICS SM Editor

Then, select which region to use as shown in Figure 6-4 on page 147.

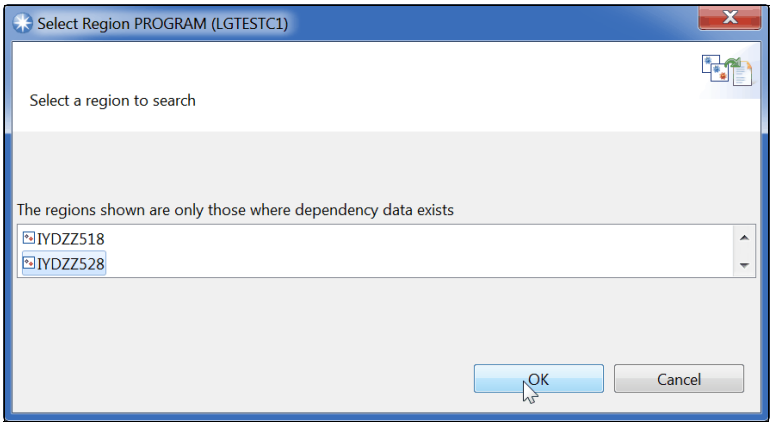


Figure 6-4 Select region IYDZZ528

The editor window for the program is now open as shown in Figure 6-5.

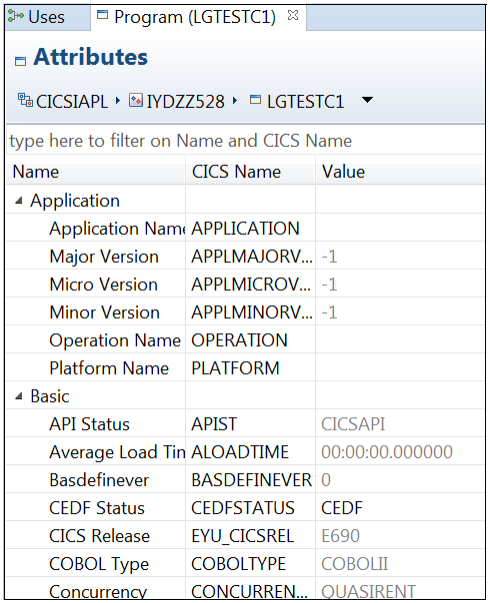


Figure 6-5 Program editor for LGTESTC1

You can now enter a value for the operation field as shown Figure 6-6. Enter **CTRL+S** or select **FILE** → **SAVE** to save the change.

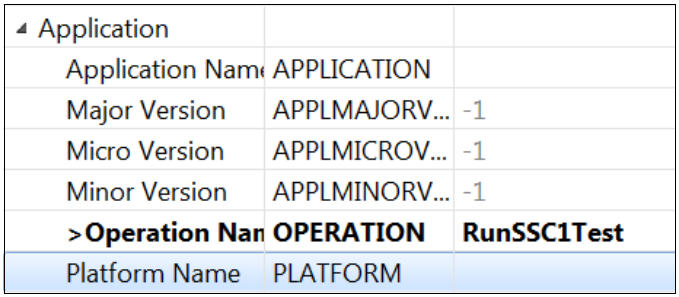


Figure 6-6 Enter an operation value

We now collect CICS IA Application data for transaction SSC1.

6.2.2 Collect CICS IA Application data for transaction SSC1

Before starting the CICS IA dependency collector, ensure that you are collecting application data. To set this option as the DEFAULT, you need to use the CINT transaction. Select option 2 for **Configure Region Options** then select option 4 for **Options** against the DEFAULT entry shown in Figure 6-7.

```
CIU200        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/19
                        Region Configuration Menu                    09:45:51AM

Type action code then press ENTER.                                   More :
1=Add Region   2=Copy Region   3=Delete Region   4=Options


     CICS      CICS   New       New
 Act Applid    Sysid  Applid    Sysid  Status       Collecting
  4  DEFAULTS  DFTS
     ALL       ALL
     IYDZZ518  Z518                  STOPPED
     IYDZZ528  Z528                  STOPPED
     IYDZZ538  Z538                  STOPPED




CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC30

F1=Help      F2=           F3=Exit      F4=          F5= Refresh  F6=
F7=Page Up   F8=Page Down  F9=          F10=         F11=         F12=
```

Figure 6-7 Default options

Select 8 for **Application Data** as shown in Figure 6-8.

```
CIU29O        CICS Interdependency Analyzer for z/OS - V5R3MO       2015/10/19
                        Resource Options for                        09:47:3OAM
                 CICS Sysid:  DFTS  CICS Applid:  DEFAULTS

Type action code then press ENTER: 8

    1 = General Options
    2 = Time/Date Options

    Interdependency Options            Affinity Options
    3 = CICS Options for APIs          7 = Affinity Options
    4 = CICS Options for SPIs
    5 = DB2/IMS/MQ/CPSM Options        CICS TS Applications
    6 = Natural Options                8 = Application Data




CICS Sysid: Z528    CICS Applid: IYDZZ528    TermID: TC3O

F1=Help      F2=           F3=Exit      F4=           F5=           F6=
F7=          F8=           F9=          F1O=          F11=          F12=Exit
```

Figure 6-8 Select Application Data

Enable the collection of Application Data by entering a **Y** as shown in Figure 6-9.

```
CIU21O        CICS Interdependency Analyzer for z/OS - V5R3MO        2015/10/19
                     Application Data Collection Options             09:49:16AM

Type Collection Option then press ENTER :

Enable collection of Application Data: Y

Selected Application:
ALL.









CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC3O

F1=Help    F2=          F3=Save&Exit  F4=              F5= Refresh  F6=
F7=        F8=          F9=           F1O=             F11=         F12=Cancel
```

Figure 6-9 Enable collection of Application Data

You also need to collect DETAILED information for transactions. Use the CINT transaction to check this value. Select option 3 for **CICS Options for APIs** from the panel shown in Figure 6-8 on page 149 and check the API values as shown in Figure 6-10.

```
CIU240        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/19
                        CICS Resources Options for                   09:53:02AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.

Detect command types: Y=Yes, N=No
                      D=Yes+Detail ( Only for API types marked with * )
APIs
*Programs . . . D *Files. . . . . D *Transactions . D  Task Control . Y
 Presentation . Y *TS Queues  . . D *TD Queues  . . D  Journals . . . Y
 DTP  . . . . . Y  Counters . . . Y  FEPI . . . . . Y *WEB Services . D
*Exits  . . . . D  Others . . . . Y *EVENTS . . . . D  ATOMServices . Y
 XMLtransform . Y  WSAddressing . Y




CICS Sysid:  Z528   CICS Applid:  IYDZZ528   TermID:  TC30

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 6-10 Collect Detail values

You can also use the IA plug-in to enable application data collection for individual regions by editing collector options against the required region in the IA Operations view as shown in Figure 6-11.

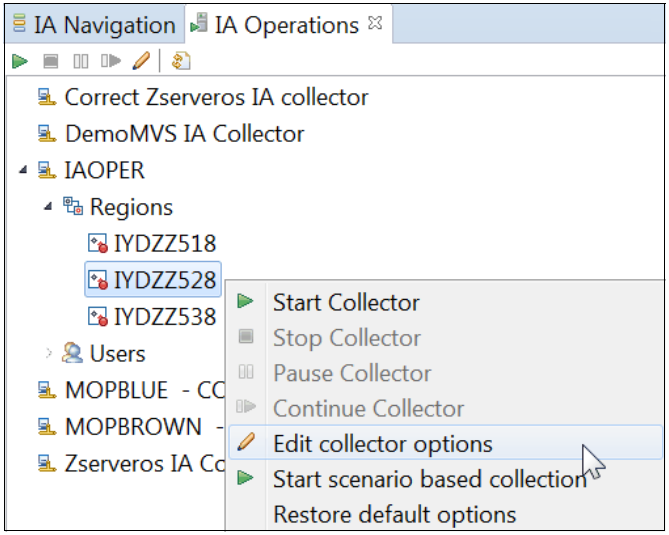


Figure 6-11 Edit collector options

We can now start the collector, run the SSC1 transaction, stop the collector, and then load the dependency data using sample job CIUUPDB with a collection ID of APPLTEST. For more information about setting the collection ID, see 2.3, “Using the collection ID to manage your data” on page 44.

We can now use the CICS IA plug-in to identify application entry points and dependencies.

6.2.3 Identify and create an entry point CICS Bundle

CICS Platform and Application data captured by CICS IA can be viewed by using the `Cloud Explorer` folder in the IA Navigation view. The data collected for a dummy application collection is saved under a reserved application name of TEST_OPERATIONS(0.0.0) as shown in Figure 6-12 on page 152.

You can see that we now have an entry for the dummy operation `RunSSC1Test`. The name of the operation is the value that we set in the operation field from the initial program `LGTESTC1`.

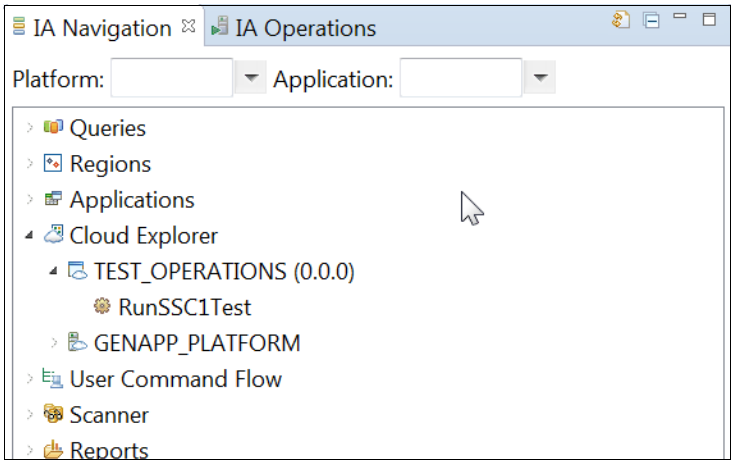


Figure 6-12 Cloud Explorer folder

There are a number of options available to you via right-click against the TEST_OPERATIONS application or against the operation RunSSC1Test as shown in Figure 6-13.

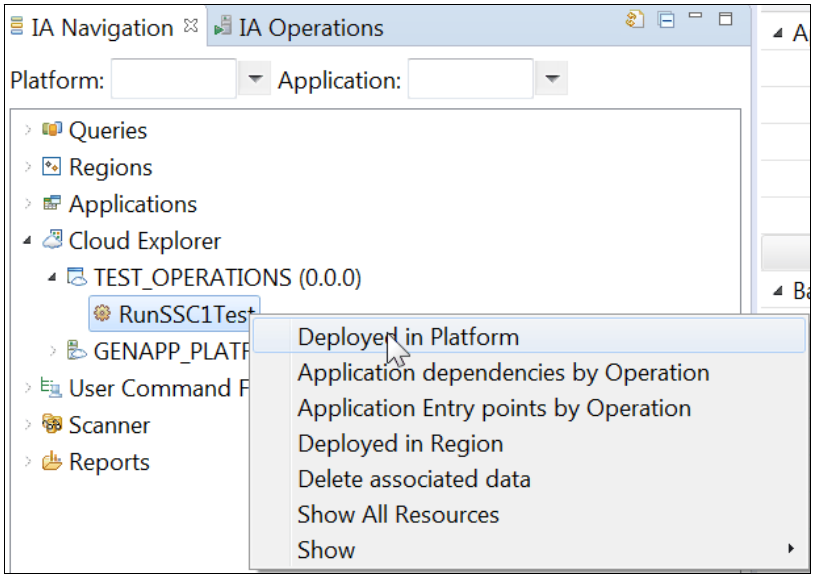


Figure 6-13 Options available for operations

We use the **Application Entry points by Operation** option to drive the Entry Point wizard as shown in Figure 6-14.

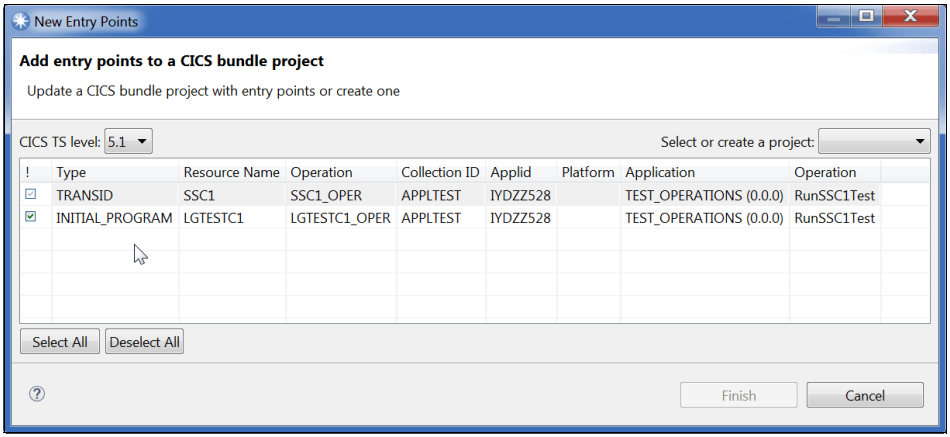


Figure 6-14 New Entry Point wizard

This wizard lists all the available entry points that you can define at CICS TS V5.3 level. We list the following entry points:

- Transactions
- Initial programs associated with a transaction
- URIMAPs associated with a CICS Webservice

The wizard allows you select at which level of CICS you are going to deploy your application as shown in Figure 6-15.

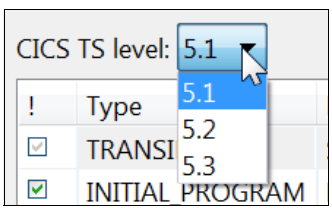


Figure 6-15 Select CICS level

The level of CICS that you select controls which entry point types you can select and create:

- V5.1: Program entry points
- V5.2: Program and URIMAP entry points
- V5.3: Transaction, program, and URIMAP entry points

We are running at CICS V5.3 so we select V5.3 and create a transaction entry point as shown in Figure 6-16.

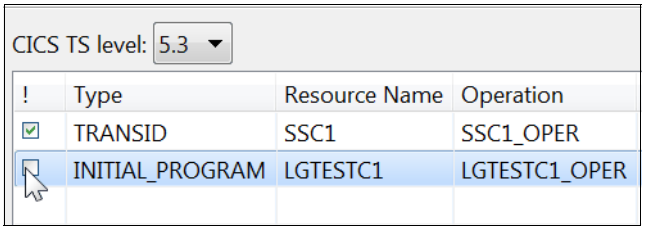


Figure 6-16 Select a transaction entry point

The wizard creates a default name for the entry point operation based on the resource name and type. In Figure 6-16, you can see that for the transaction entry point this default is set to SSC1_OPER. The wizard allows you to edit this value. We change it to `Genapp_EP1` as shown in Figure 6-17.

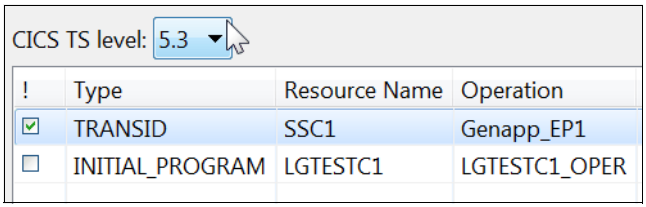


Figure 6-17 Edit the Operation name

We are ready to create or update a CICS Bundle project. To create a new CICS Bundle project, select the **New project** option as shown in Figure 6-18.

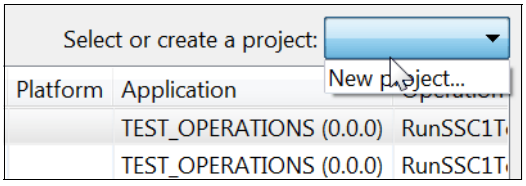


Figure 6-18 Create new project

This action drives the CICS Explorer wizard to create a CICS Bundle project. Enter a meaningful name as shown in Figure 6-19.

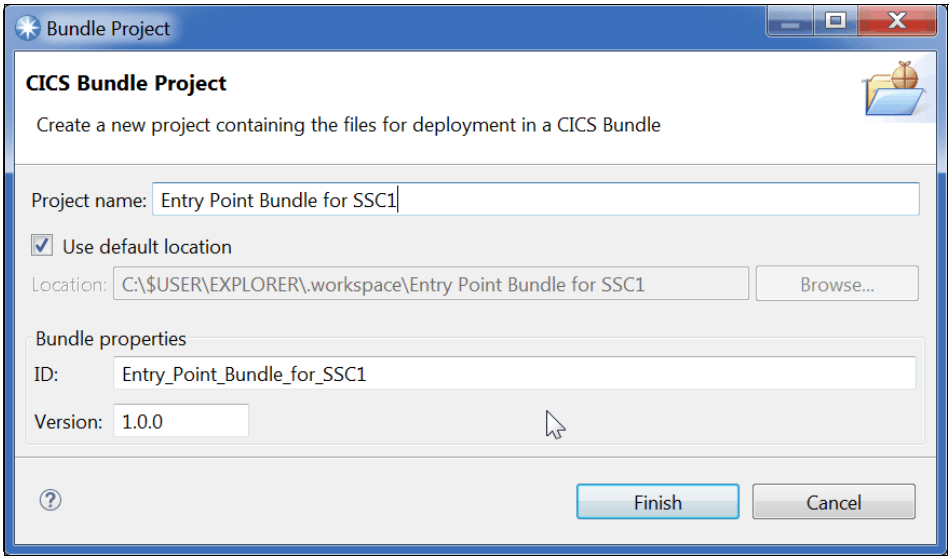


Figure 6-19 CICS Bundle project

Click **Finish** to create the project and open the editor as shown in Figure 6-20.

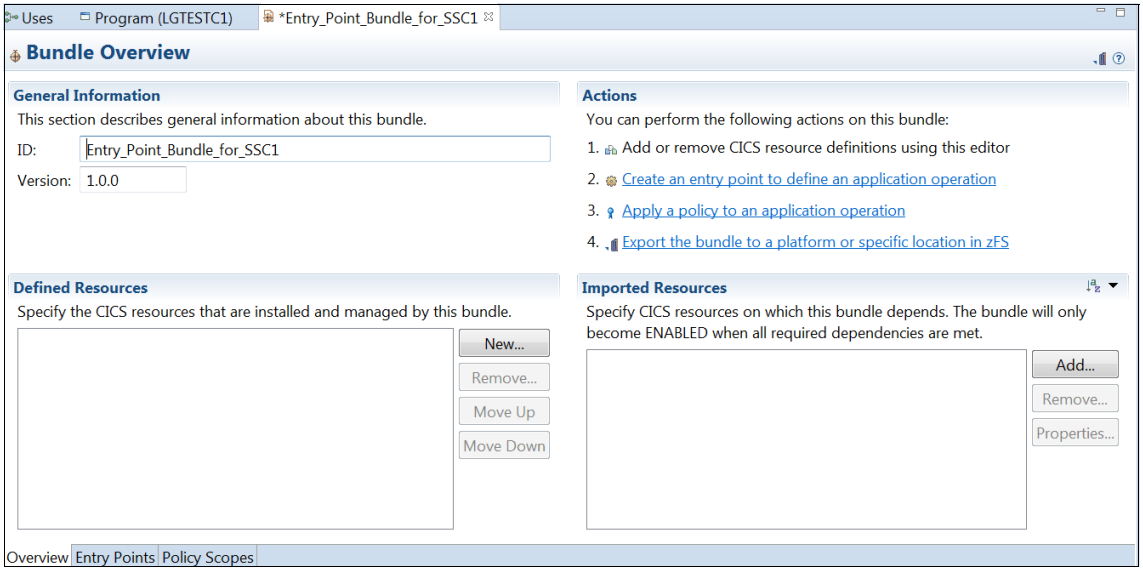


Figure 6-20 Bundle project editor

Click **Finish** in the New Entry Points window shown in Figure 6-14 on page 153 to add the transaction as an entry point to the project as shown in Figure 6-21.

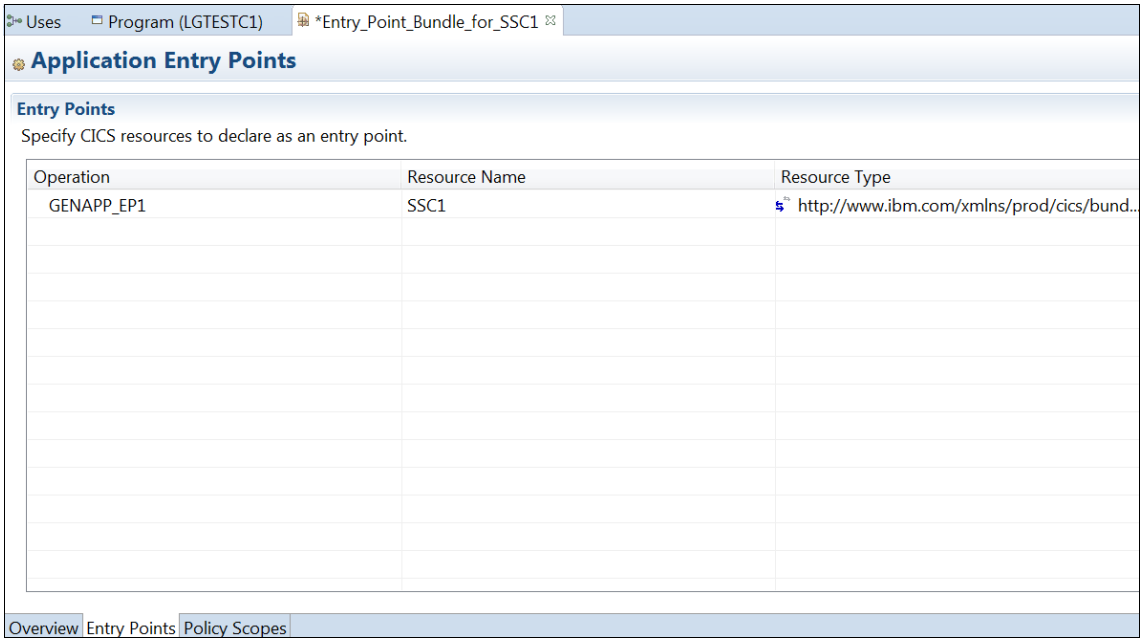


Figure 6-21 Transaction SSC1 entry point

You have now created a CICS Bundle containing our entry point. This bundle can be included when creating your CICS Application Bundle as shown in Figure 6-22.

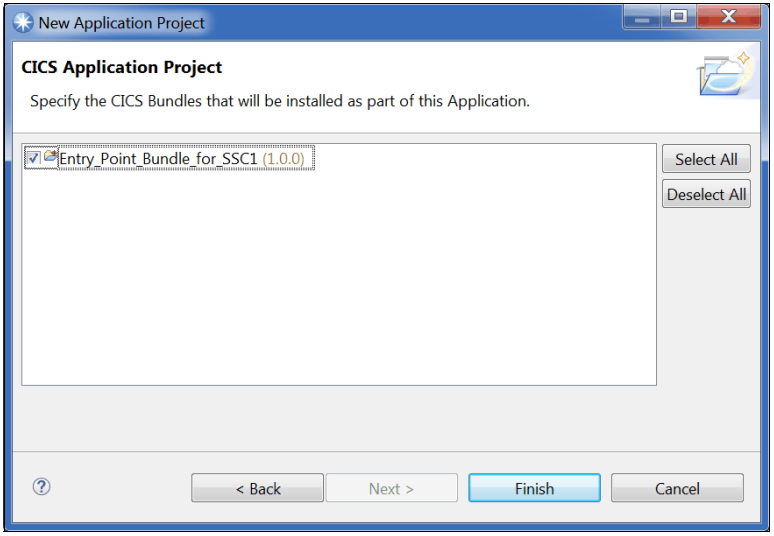


Figure 6-22 Creating an application project

6.2.4 Identify and create a dependency CICS Bundle

Right-click your test operation and select the **Application dependencies by Operation** option, as shown in Figure 6-23 on page 158.

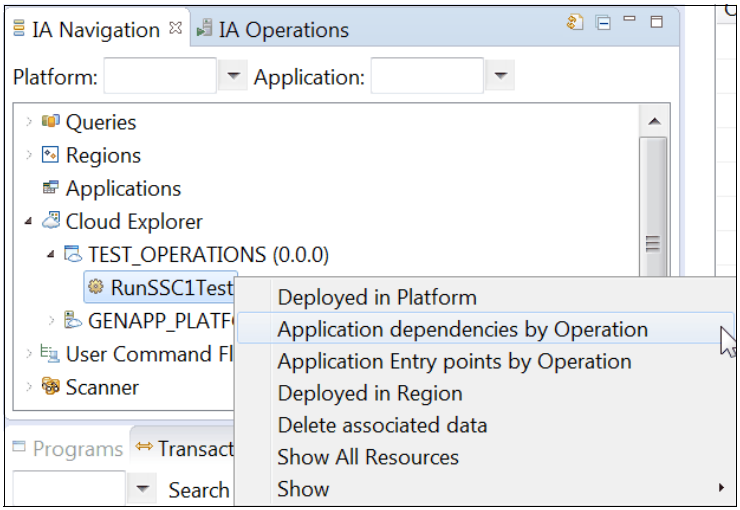


Figure 6-23 Application dependencies by Operation selection

This action drives the **New Dependency** wizard as shown in Figure 6-24.

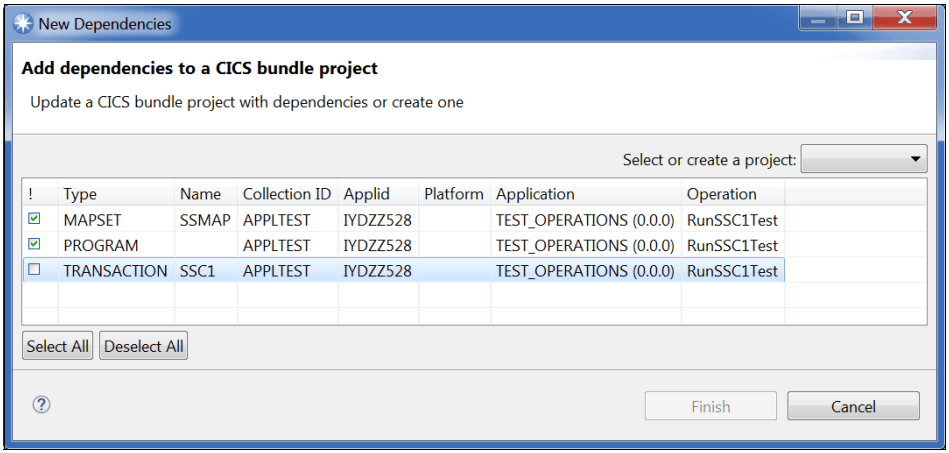


Figure 6-24 New Dependency wizard

This wizard lists all possible dependencies that we have collected data for and that can be defined as a dependency in a CICS Bundle project.

We report the following CICS dependencies:

- TRANSID
- PROGRAM
- FILE

- MAPSET
- ATOMSERVICE
- DOCTEMPLATE
- TDQUEUE
- JVMSERVER
- LIBRARY
- TSMODEL
- ENQMODEL
- JOURNALMODEL
- PARTITIONSET
- PIPELINE
- PROCESSTYPE
- TCPIPSERVICE
- URIMAP
- EVENTBINDING
- EPADAPTER
- EPADAPTERSET
- WEBSERVICE
- XMLTRANSFORM

Note: The CICS IA does not collect information about many of these resource types unless they are used in an SPI command such as **`INQUIRE TSMODEL`**.

We also report if there is a dependency on a DB2 connection and an IBM MQ connection. For the DB2 connection, we report the DB2 subsystem ID but not the DB2 connection definition name. For IBM MQ, we report that the application or operation has used IBM MQ resources.

In our example, we add the MAPSET and PROGRAM as a dependency as shown in Figure 6-24 on page 158. To create a new CICS Bundle project, select the **New project** option as shown in Figure 6-25.

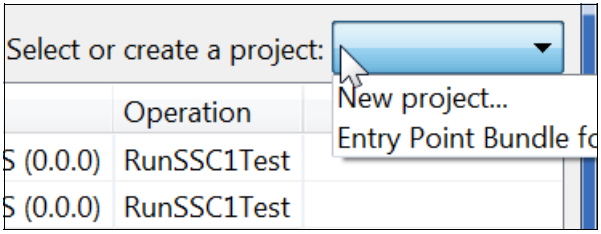


Figure 6-25 New project options

This action drives the CICS Project wizard. Enter a meaningful name as shown in Figure 6-26.

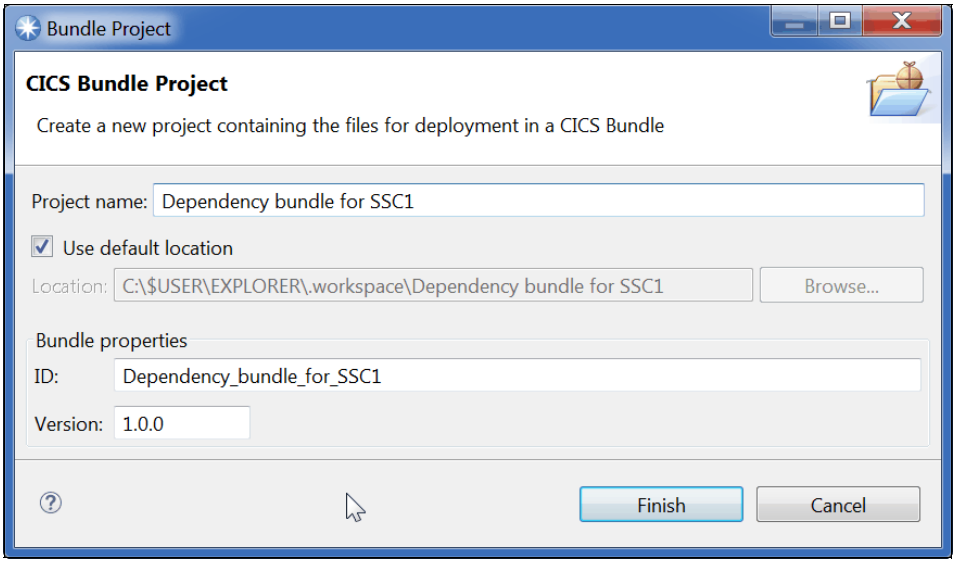


Figure 6-26 CICS Bundle project

Click **Finish** to create the project.

Click **Finish** in the New Dependency window as shown in Figure 6-24 on page 158 to add the dependencies to the project as shown in Figure 6-27.

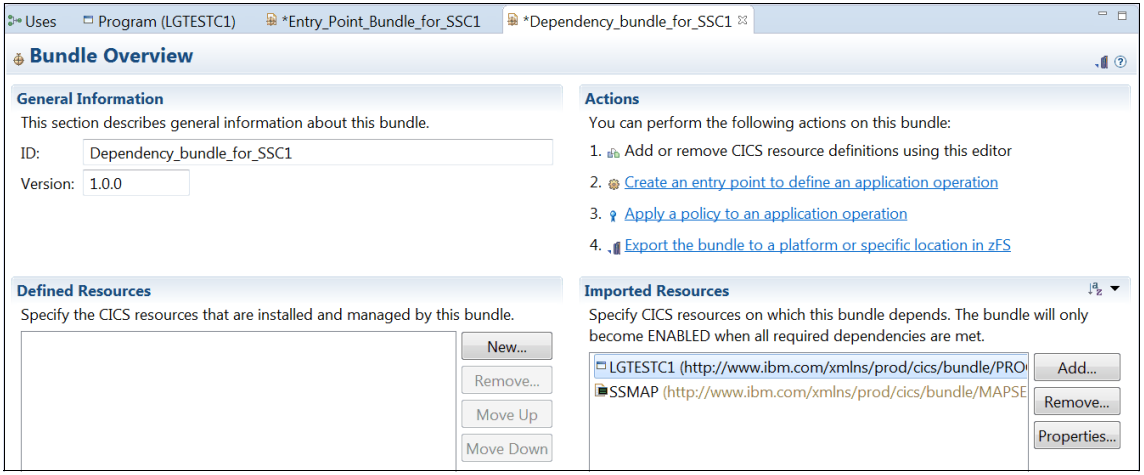


Figure 6-27 Dependency bundle

We have now created a CICS Bundle containing our dependencies. Again, this bundle can be included when creating your CICS Application Bundle as shown Figure 6-28.

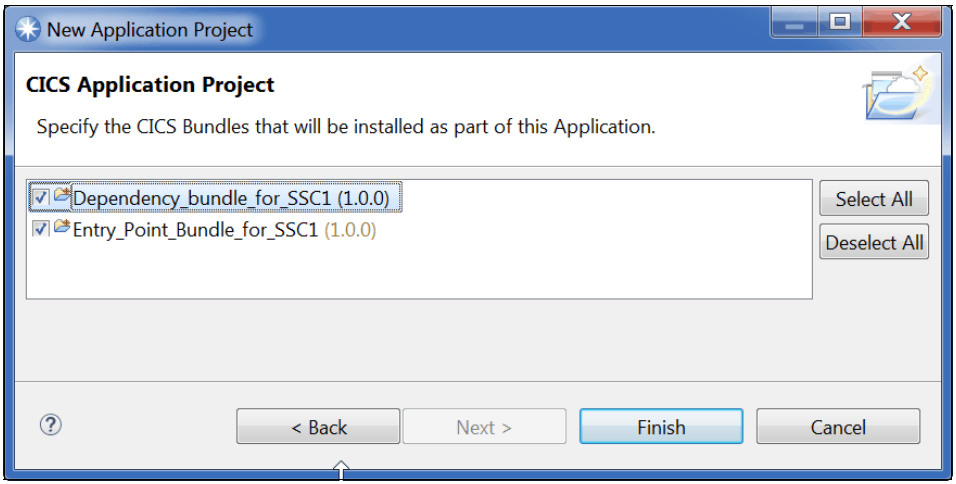


Figure 6-28 CICS Application: CICS Bundle selection

6.3 Identifying entry points and dependencies: Two-phase approach

In this section, we use existing CICS IA data to first identify entry points and create a CICS Bundle project to contain these entry points. The CICS Bundles can then be used to create an application. The application can then be deployed in to a platform. We can then collect and load CICS IA Application data. We can then use this data to identify the application dependencies.

We perform the following steps:

- Identify entry points by resource used
- Identify entry points by using the same resource type
- Identify dependency for an application

6.3.1 Identify entry points by resource used

The CICS IA plug-in provides a wizard to identify entry points by a specific resource used. You select to group at the transaction level, the program level, or a Webservice level.

For the program and transaction grouping, we return the following data:

- The transactions using the resource
- The initial program for the transaction using the resource
- The actual programs using the resource (BACK_PROG)

For Webservices grouping, we return the following data:

- The Webservices using the resource
- The programs associated with the Webservices using the resource
- The URIMAPs associated with the Webservices using the resource
- The actual programs using the resource (BACK_PROG)

In our example, we identify entry points grouped by transaction that use the FILE resource KSDSPOLY.

Right-click the resource and select as shown in Figure 6-29. This option can be driven against any resource type in any view in the CICS IA plug-in.

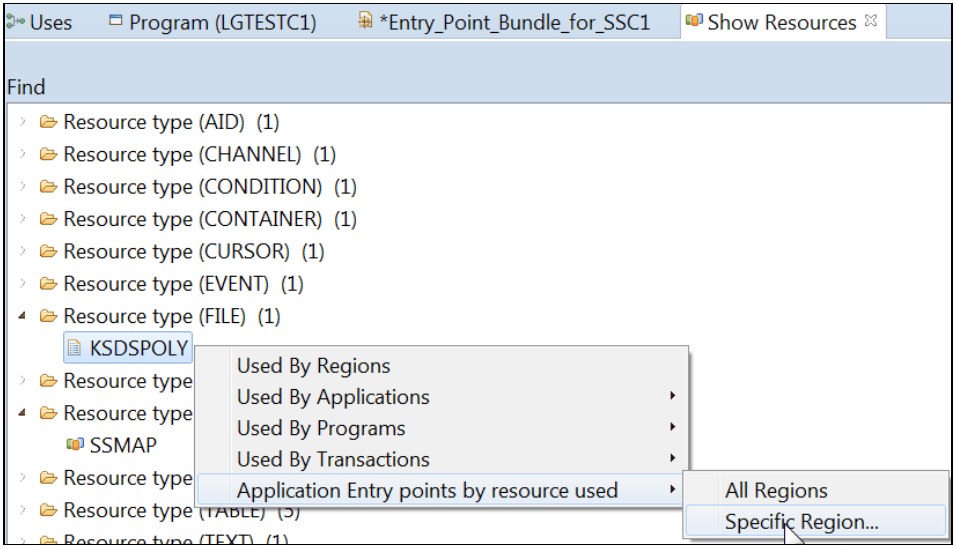


Figure 6-29 Application entry points by resource used

We select region IYDZZ528. The New Entry Point wizard is driven as shown in Figure 6-30 on page 163.

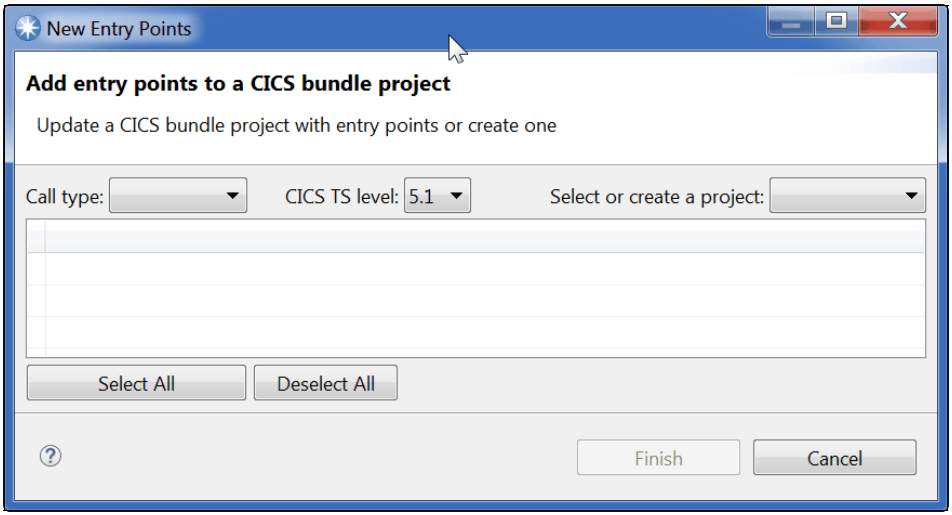


Figure 6-30 New entry points

To choose your grouping, use the **Call type** option as shown in Figure 6-31.

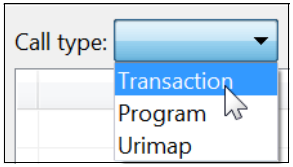


Figure 6-31 Select Call type

We select the **Transaction** call type. The wizard is now populated with possible entry points as shown in Figure 6-32 on page 164.

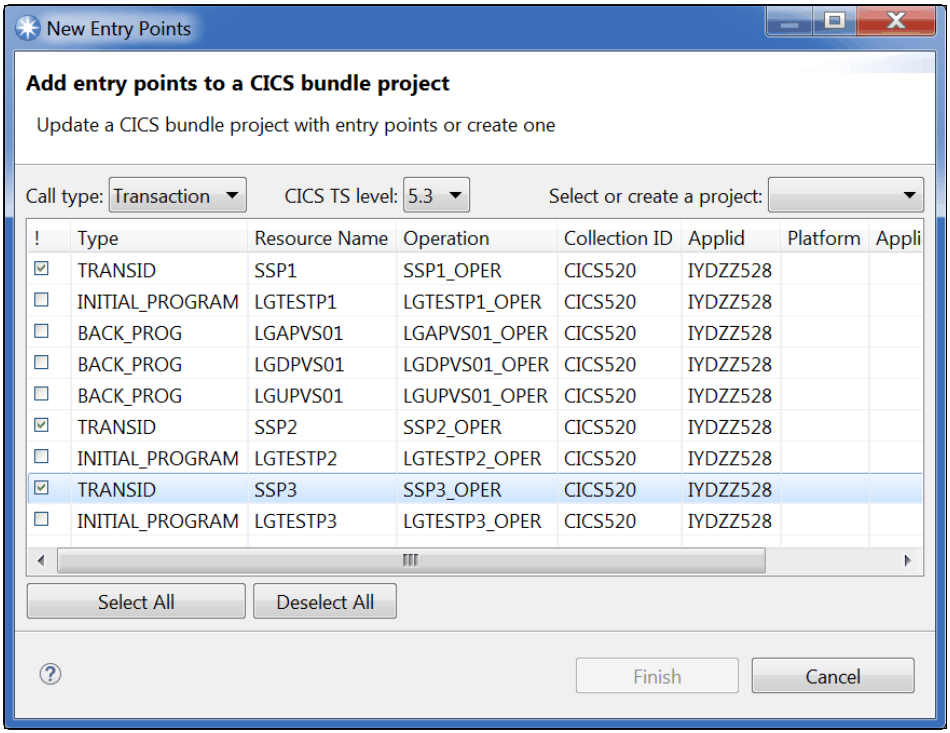


Figure 6-32 New entry points by transaction

The wizard allows you to select your CICS release as shown in Figure 6-15 on page 153.

The level of CICS that you select controls which entry point types you can select and create:

- V5.1: Program entry points
- V5.2: Program and URIMAP entry points
- V5.3: Transaction, program, and URIMAP entry points

We create transaction entry points at CICS V5.3 and we use the default operation names generated by CICS IA, as shown in Figure 6-32.

To create a new CICS Bundle project, select the **New project** option as shown in Figure 6-33.

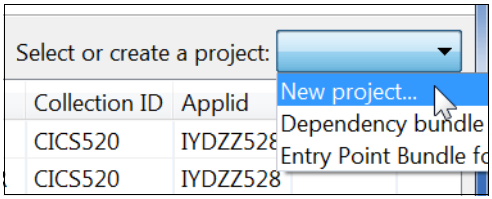


Figure 6-33 Create a new project

This action drives the CICS Explorer CICS Bundle Project wizard as shown in Figure 6-34.

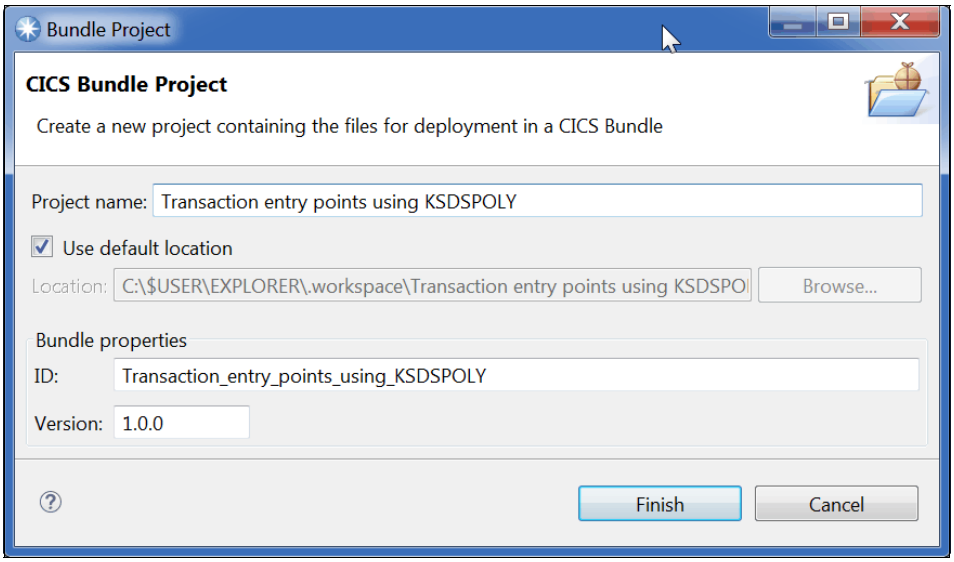


Figure 6-34 CICS Bundle Project wizard

Enter a meaningful name and click **Finish** to open the CICS Bundle editor as shown in Figure 6-35 on page 166.

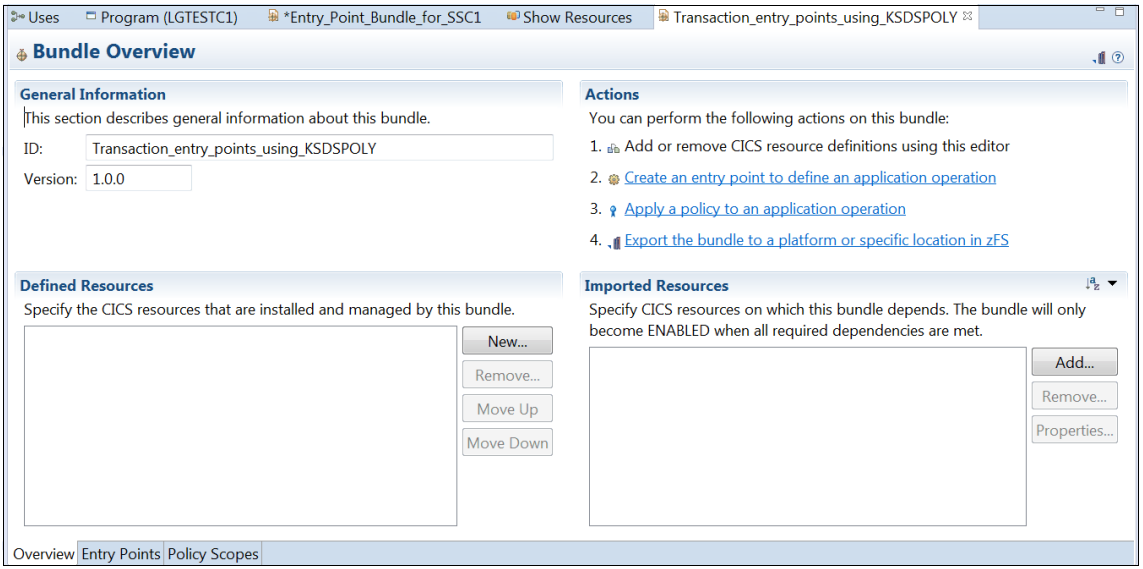


Figure 6-35 CICS Bundle editor

Click **Finish** in the **New Entry Point** wizard as shown in Figure 6-32 on page 164 and the entry points are added to your project as shown in Figure 6-36.

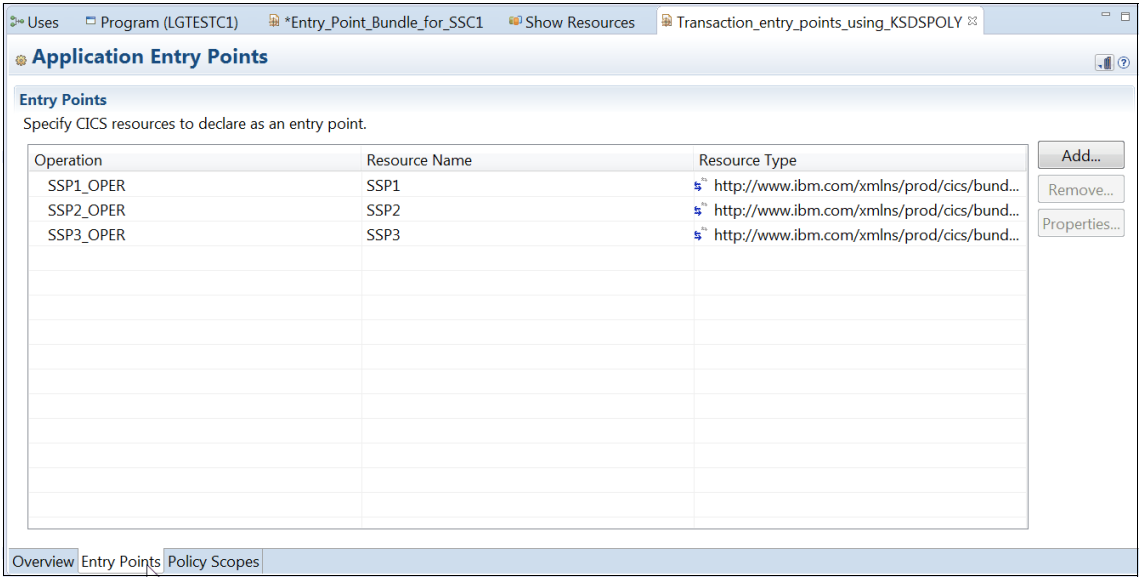


Figure 6-36 CICS Bundle entry points

6.3.2 Identify entry points that use the same resource type

In this section, we identify entry points for a transaction, program, or Webservices that use the same resource type.

For the program and transaction, we return the following data:

- The transactions using the resource
- The initial program for the transaction using the resource
- The actual programs using the resource (BACK_PROG)

For Webservices, we return the following data:

- The Webservices using the resource
- The programs associated with the Webservices using the resource
- The URIMAPs associated with the Webservices using the resource
- The actual programs using the resource (BACK_PROG)

In our example, we start with transaction SSC1 and identify entry points for all other transactions that use a resource type of MAPSET.

Right-click transaction SSC1 and select **Application Entry points by transaction resource usage** as shown Figure 6-37.

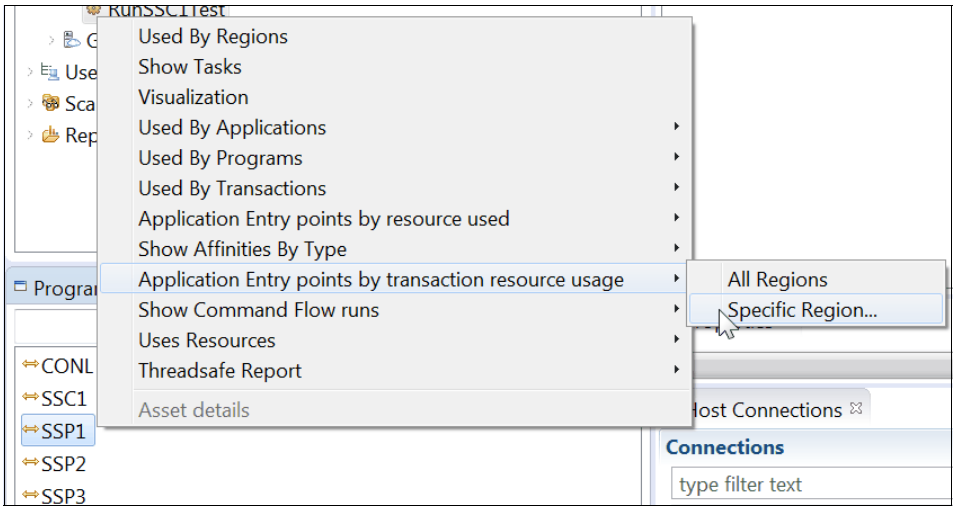


Figure 6-37 Application entry points by resource usage

We select region IYDZZ528. The **New Entry Point** wizard is driven as shown in Figure 6-38 on page 168.

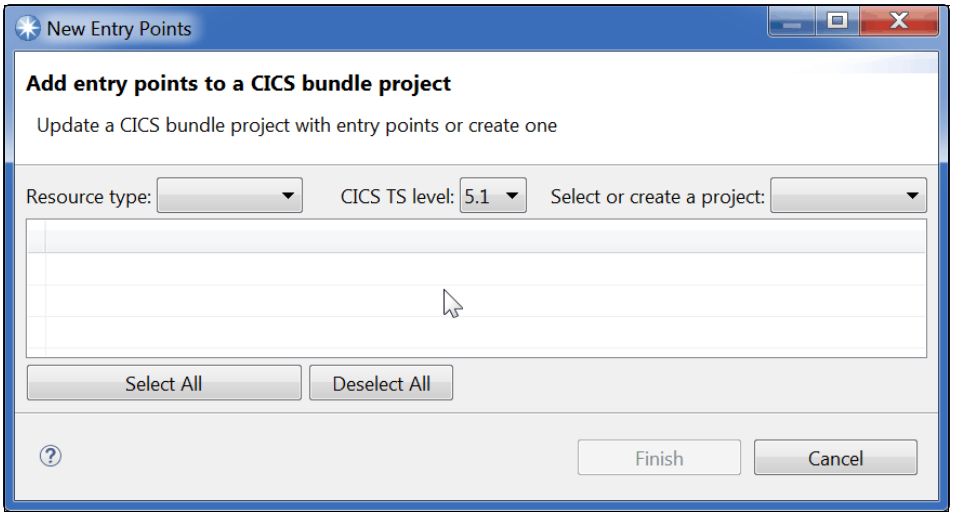


Figure 6-38 New Entry Point wizard

To select the **MAPSET** as the resource type, select the **Resource types** options as shown in Figure 6-39.

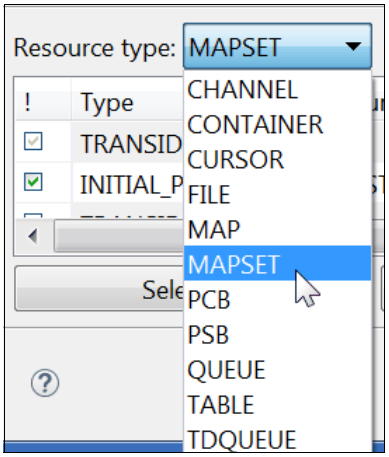


Figure 6-39 Select MAPSET

The wizard is now populated by the transactions and initial programs that use a resource type of MAPSET. In this example, we create PROGRAM entry points as we are on CICS TS V5.2 where TRANSACTON entry points are not available.

We selected all programs as shown in Figure 6-40.

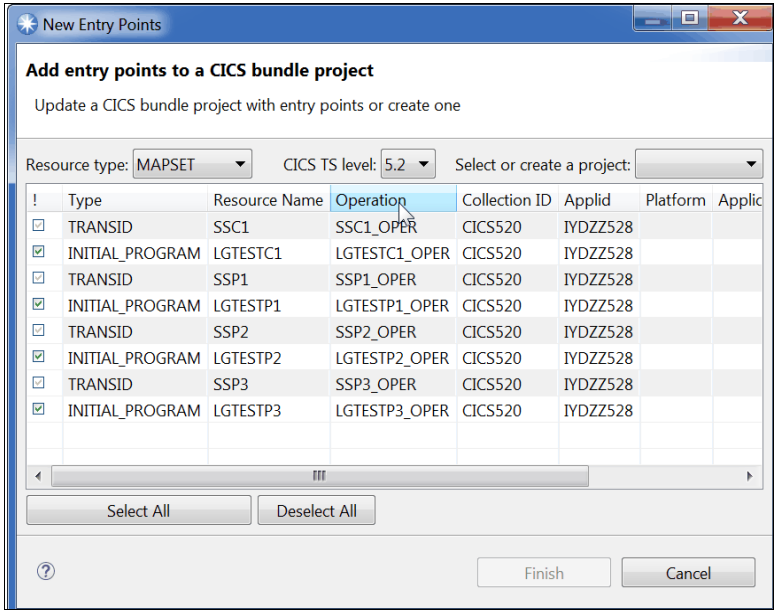


Figure 6-40 Select PROGRAM entry points

We can now create a new project by following similar steps to what we did in 6.3.1, “Identify entry points by resource used” on page 161.

Figure 6-41 shows the CICS Bundle project that we created containing program entry points.

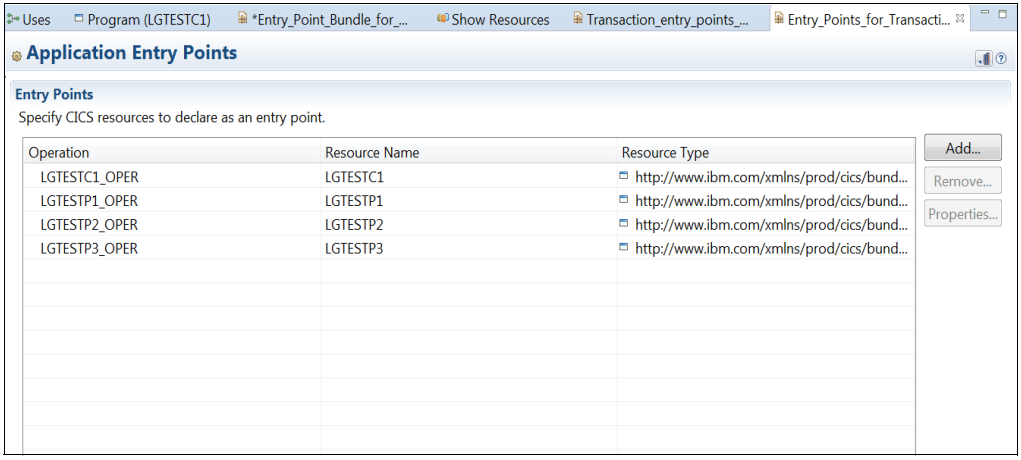


Figure 6-41 Program entry points

6.3.3 Identify dependencies by application

After we created and deployed applications using the CICS *entry point* Bundles created in the previous steps, we can collect and load CICS IA Application data as described in 6.2.2, “Collect CICS IA Application data for transaction SSC1” on page 148.

Then, we can proceed to the second phase and identify the resource dependencies for the application. In this example, we use the GENAPP_APPL shipped as IVP data with CICS IA.

Right-click the application and select **Application dependencies by Application** as shown in Figure 6-42.

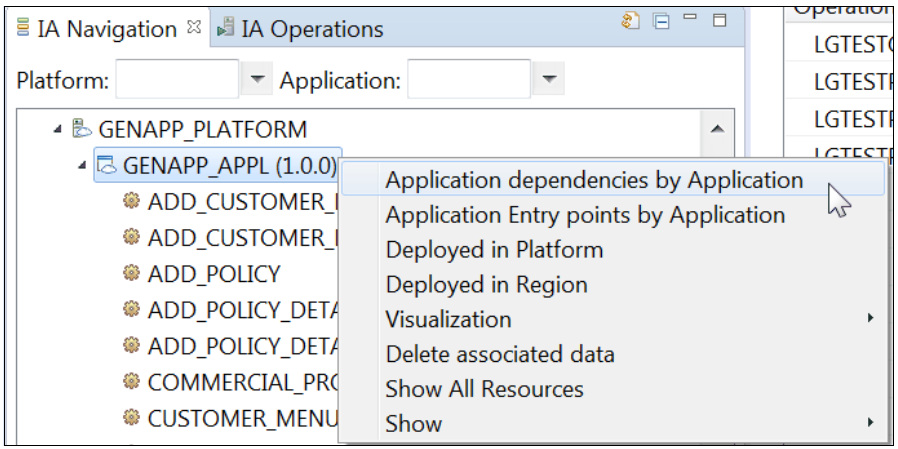


Figure 6-42 Application dependencies by application

This action drives the **New Dependency** wizard as shown in Figure 6-43.

Details about how this wizard works are described in 6.2.4, “Identify and create a dependency CICS Bundle” on page 157.

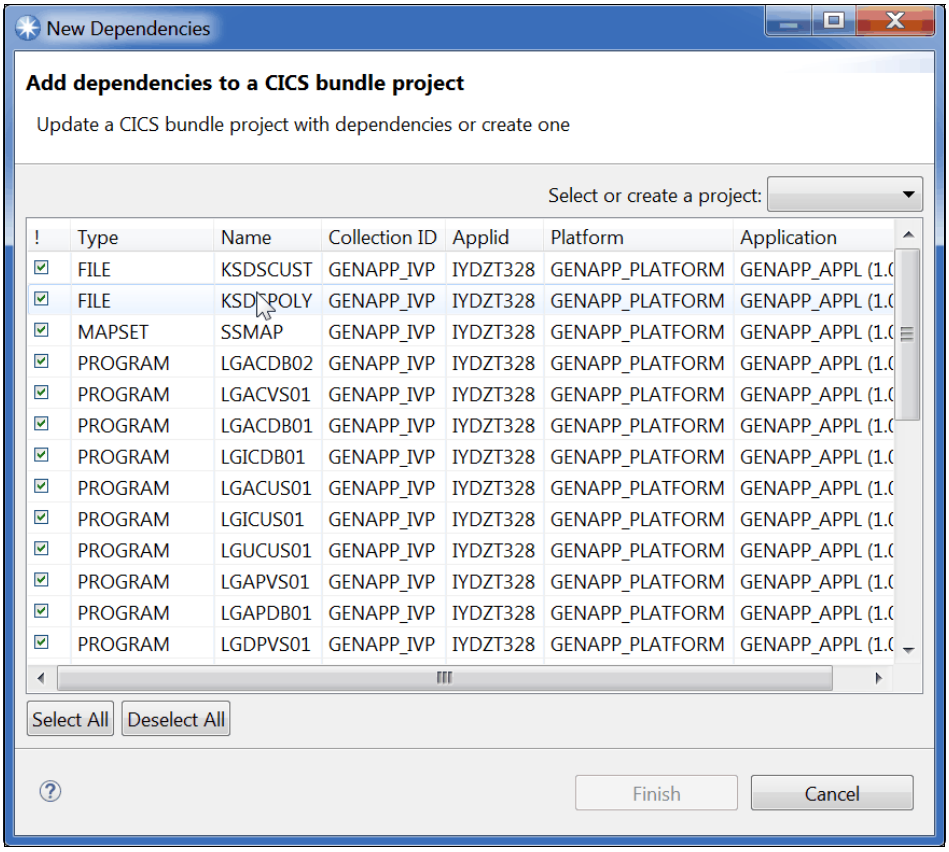


Figure 6-43 Add dependencies to CICS Bundle

The DB2 entry returns the DB2 subsystem ID and not the actual DB2 resource definition name. To get the definition name, use the CICS Explorer and look for the active DB2 connections as shown in Figure 6-44.

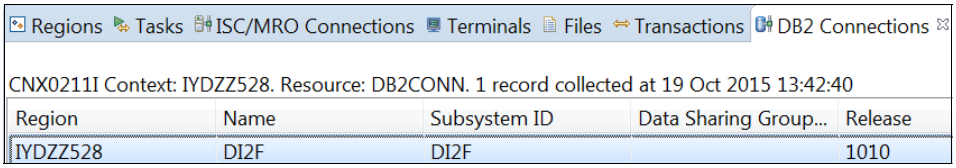


Figure 6-44 DB2 Connections view in the CICS Explorer

In this example, we create a CICS Bundle for a DB2 connection dependency. In the wizard, clear all the resources, select only the DB2 entry, and select **New project** as shown in Figure 6-45.

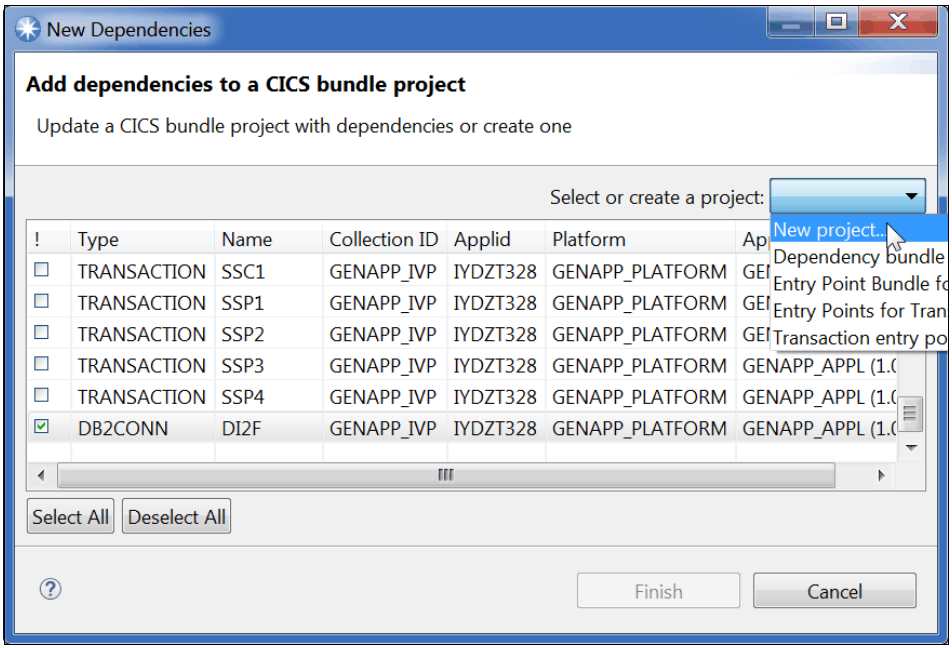


Figure 6-45 New CICS Bundle project for DB2 dependency

This drives the CICS Bundle Project wizard. Enter a meaningful name and click **Finish** as shown in Figure 6-46.

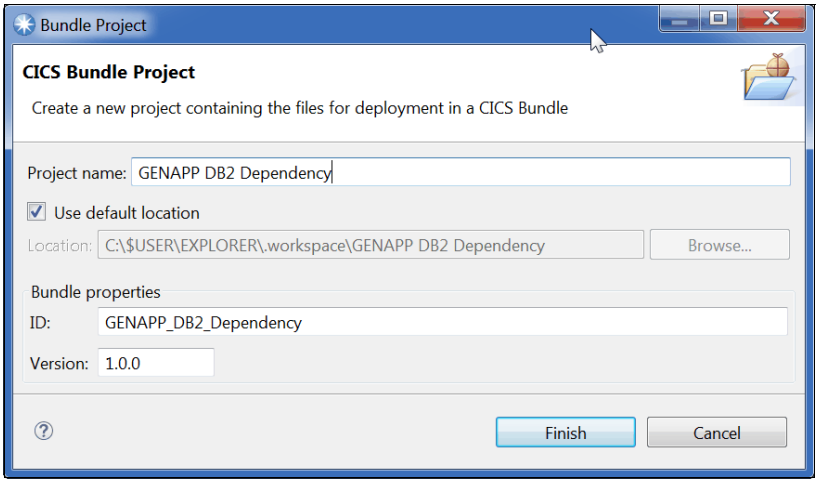


Figure 6-46 CICS Bundle project for DB2 dependency

Click **Finish** in the New Dependency wizard shown in Figure 6-45 on page 172 and the CICS Bundle project editor contains your DB2 dependency as shown in Figure 6-47.

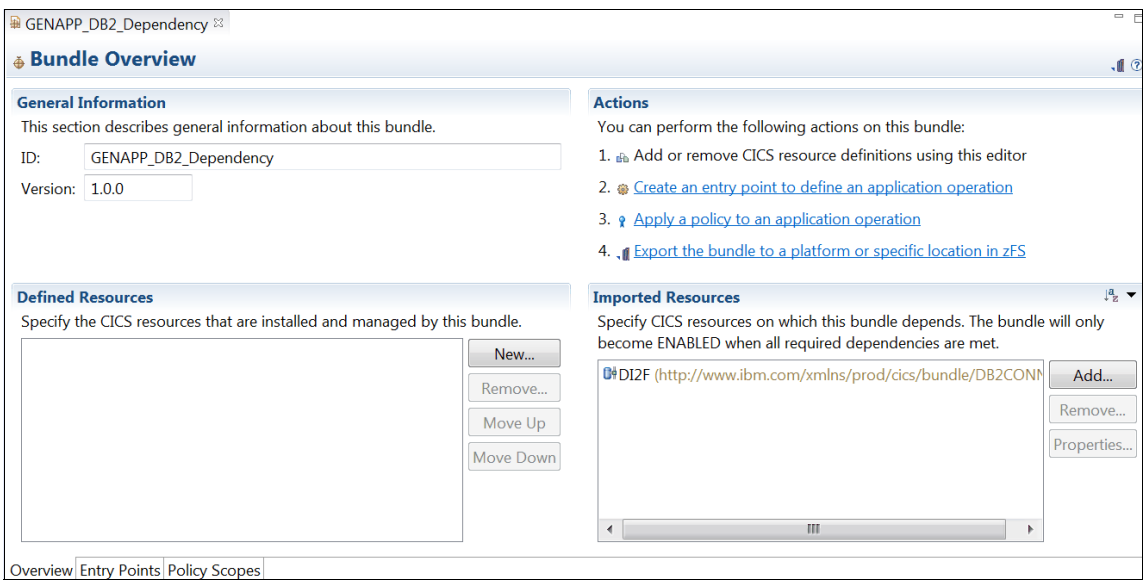


Figure 6-47 CICS Bundle project editor for GENAPP_DB2_Dependency

This CICS Bundle can now be included in a CICS Application Bundle. Alternatively, it could be added to the CICS Application Binding Bundle or the CICS Platform Bundle as a dependency.

6.4 View application data

In this section, we look at two ways of viewing resources used by an application. We use the data collected for the GENAPP_APPL application that is shipped as IVP data with the CICS IA product.

CICS IA platform and application data collected by CICS IA are displayed in the `Cloud Explorer` folder in the IA Navigation view as shown in Figure 6-48.

There are a number of options available to you, including:

- Show all resources
- Visualization

We look at these two in more detail.

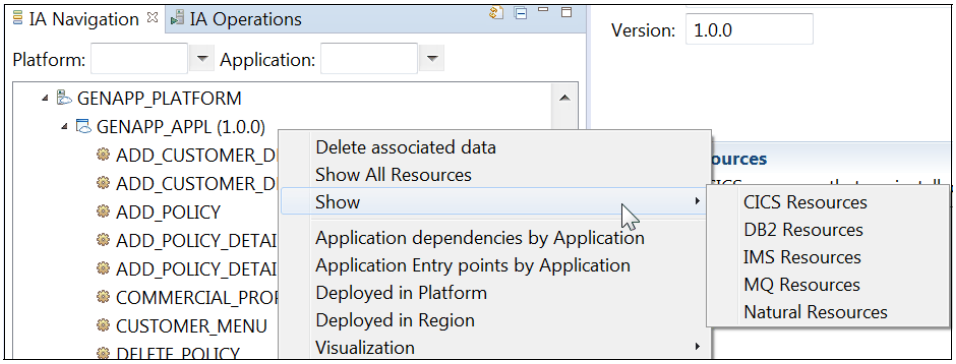


Figure 6-48 Cloud Explorer folder in the IA Navigation view

6.4.1 Show all resources for an application

Right-click the GENAPP_APPL and select **Show All Resources** as shown in Figure 6-48. This opens the Show Resources view as shown in Figure 6-49 on page 175.

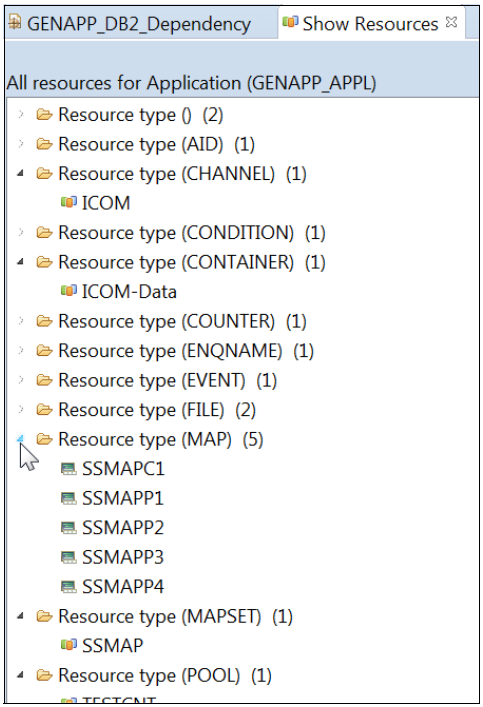


Figure 6-49 Resources used by application

6.4.2 Visualization for an application

Right-click the GENAPP_APPL and select **Visualization → Selected Application** as shown in Figure 6-50.

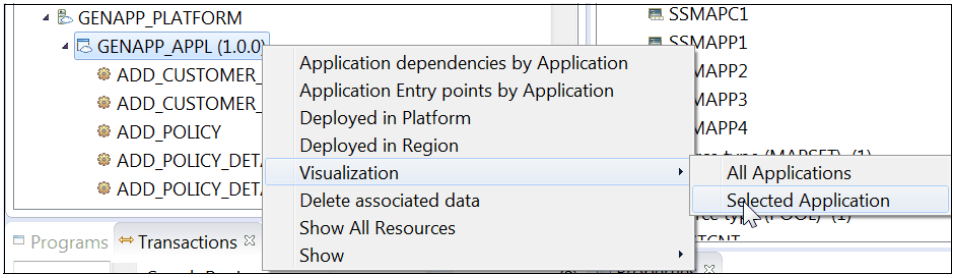


Figure 6-50 Application visualization

This opens the Resource Application view as shown in Figure 6-51.

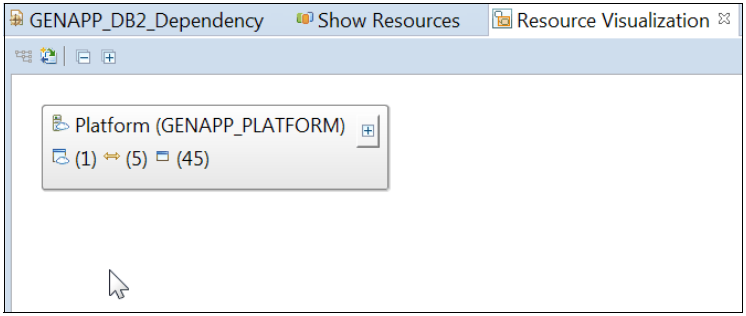


Figure 6-51 GENAPP_APPLICATION

Use the “+” icon to expand the view as shown in Figure 6-52.

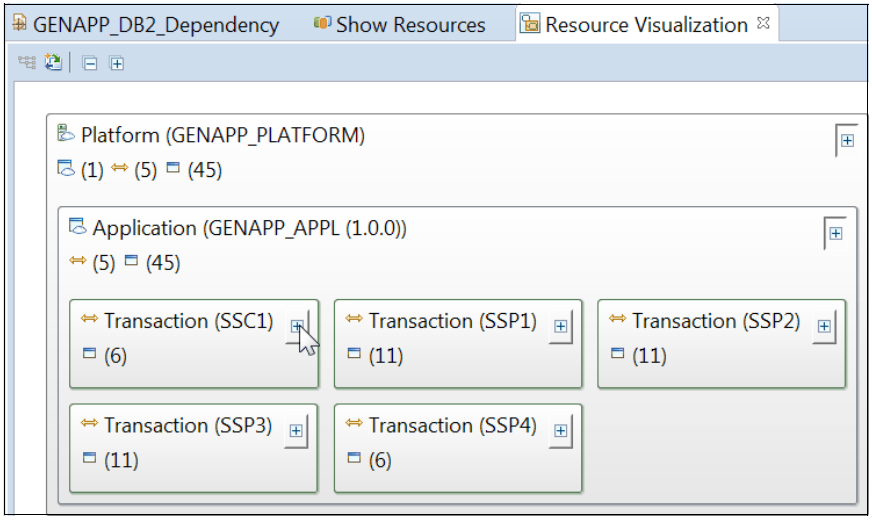


Figure 6-52 Expanded view

You can further expand the view for transaction SSC1 by using the “+” icon as shown in Figure 6-53.

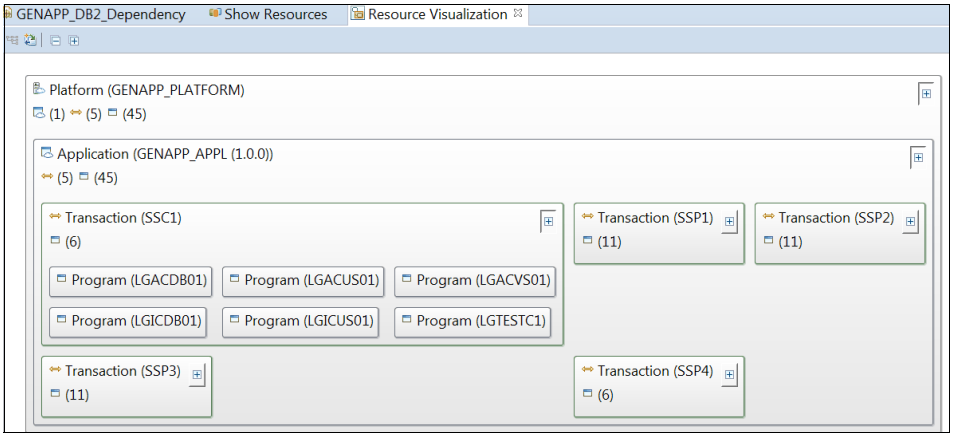


Figure 6-53 Expand transaction SSC1 to show the programs

To show the resource and connections used by transaction SSC1, right-click the transactions and select **Show Connections** as shown in Figure 6-54.

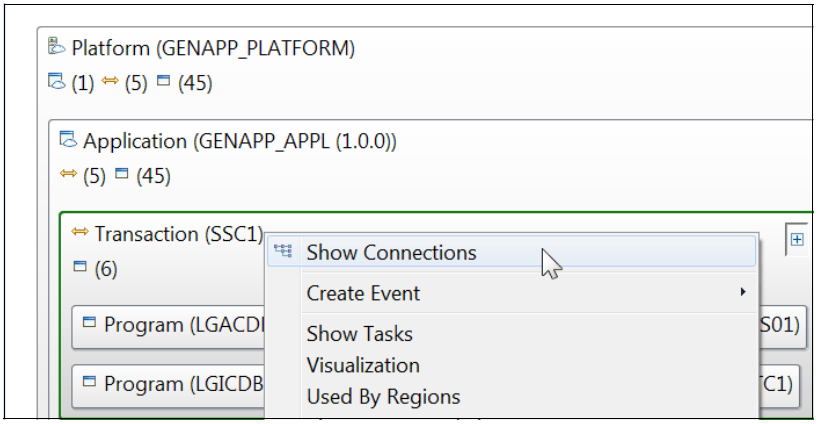


Figure 6-54 Show connections

This action opens the Connections view as shown in Figure 6-55.

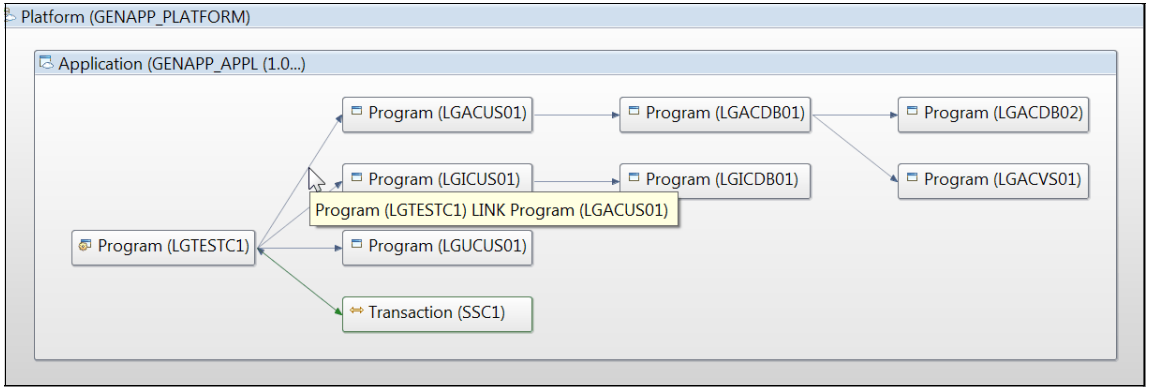


Figure 6-55 Connections view for transaction SSC1

If you hover the mouse over the connection, it shows you the command, in this case a program LINK.

You can add other resource types by switching on the filter option as shown in Figure 6-56.

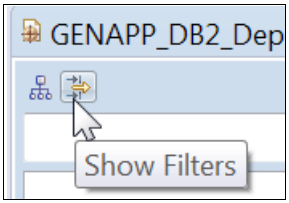


Figure 6-56 Show Filters option

You can then select which resources to display. In the example shown in Figure 6-57, we selected to filter on DB2 resources.

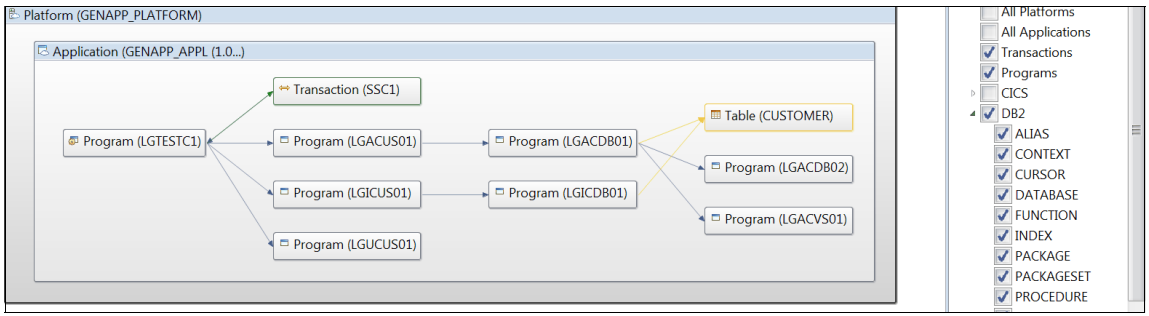


Figure 6-57 Filter on DB2 resources

You can change the orientation of the Connections view from horizontal to vertical and vice versa by using the Orientation icon as shown in Figure 6-58.

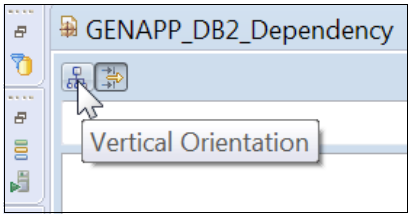


Figure 6-58 Orientation icon

7

Modernization with CICS Events

In this chapter, we demonstrate how the CICS Interdependency Analyzer (IA) collected data can assist with generating CICS Events.

7.1 Modernizing with CICS Events

In this section, we look at how CICS IA can be used to assist when creating IBM CICS Transaction Server (TS) Events.

IBM CICS TS business applications are the main source of business information in most large enterprises. The CICS run time detects instances of events that are enabled and captures the events and payload without the need to make application code changes or to provide system code. CICS Event processing is a core component of the CICS run time and provides all the qualities of service you would expect of CICS. It is possible to emit events in formats that are suitable for use by IBM Operational Decision Manager Events, IBM Business Monitor, and other users.

For more information about CICS TS Events, see the IBM Redbooks publication *Event Processing with CICS*, SG24-7792. See the section for information about "Help from IBM" on page 240.

We look at the following topics:

- Creating a CICS TS Event using the IA plug-in
- Capturing CICS IA Event data

We use the following scenario.

The GENAPP application allows you to create new insurance policies for an existing customer. It also allows you to create new customer records. The company manager wants to quickly inform his team of insurance brokers that a new customer account has been created.

Updating the GENAPP code to achieve this requirement would be expensive. By using the application knowledge provided by the collected CICS IA data, we can achieve this requirement by using a CICS TS Event.

In this example, we call the event: NEW_CUSTOMER.

7.1.1 Creating a CICS TS Event using the IA plug-in

In this section, we perform the following actions:

- Create a CICS Bundle project
- Understand how GENAPP creates a new account
- Drive the Event wizard from the CICS IA perspective

We do not deploy the event. Information about how to fully install and deploy the event can be found in “Chapter 6 - Capture Application Events” in the Redbooks publication *Event Processing with CICS*, SG24-7792. See information about “Help from IBM” on page 240.

Create a CICS Bundle project

CICS TS Events are created as CICS Bundles and are eventually exported to the zFS directory as part of the deployment. Before using CICS IA to create an event, we need to create a CICS Bundle project to manage this event.

You create a CICS Bundle project by using the CICS Explorer. Go the “Resources” perspective and right-click the Project Explorer view and select **New → Project** as shown in Figure 7-1.

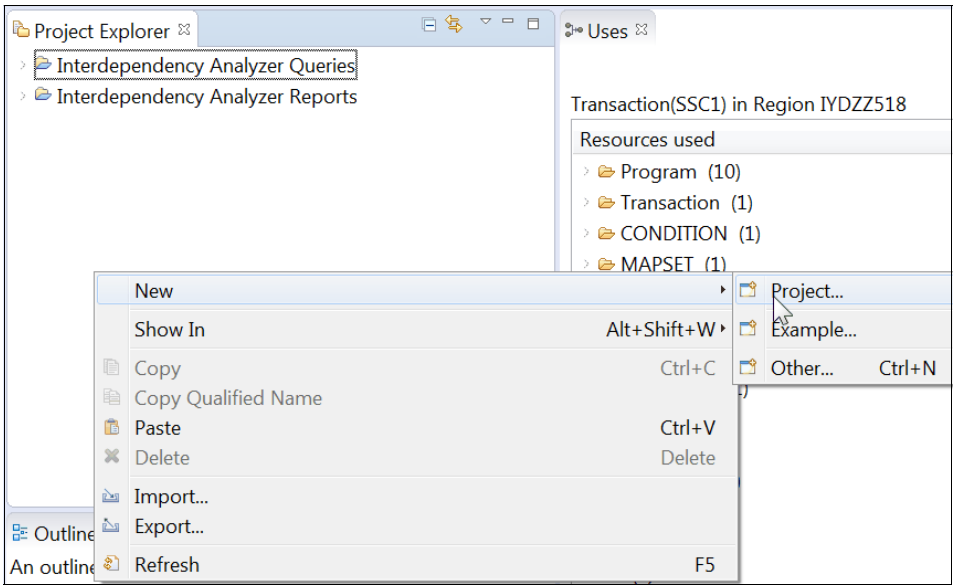


Figure 7-1 Creating a CICS Bundle project

This action opens the New Project wizard. Select “CICS Bundle Project,” as shown in Figure 7-2.

Click **Next**.

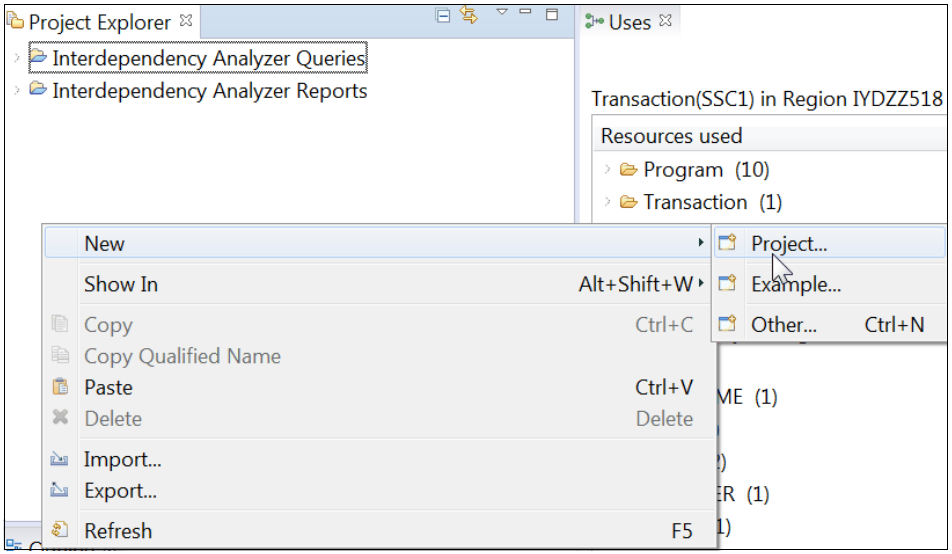


Figure 7-2 Select a CICS Bundle project

Enter a meaningful name for your CICS Bundle as shown in Figure 7-3 on page 185.

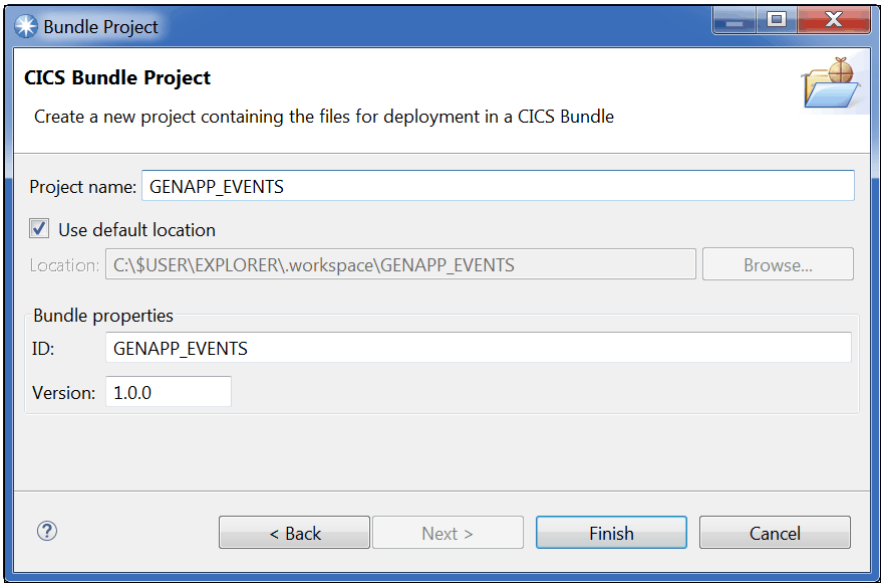


Figure 7-3 Enter a name for your CICS Bundle project

Click **Finish**. We can now go back to the IA perspective and see where we should add the event.

Where does the GENAPP application create a new account

We now need to understand the GENAPP application to determine where best to trigger the NEW_CUSTOMER event:

- Where do we start?
- What do we already know?

We know that transaction SSC1 is used to manage the customer accounts. So we see what resources are used by this transaction in region IYDZZ518, our test region.

In the Transactions view, right-click transaction SSC1 and select **Uses Resources → Specific Region** as shown in Figure 7-4.

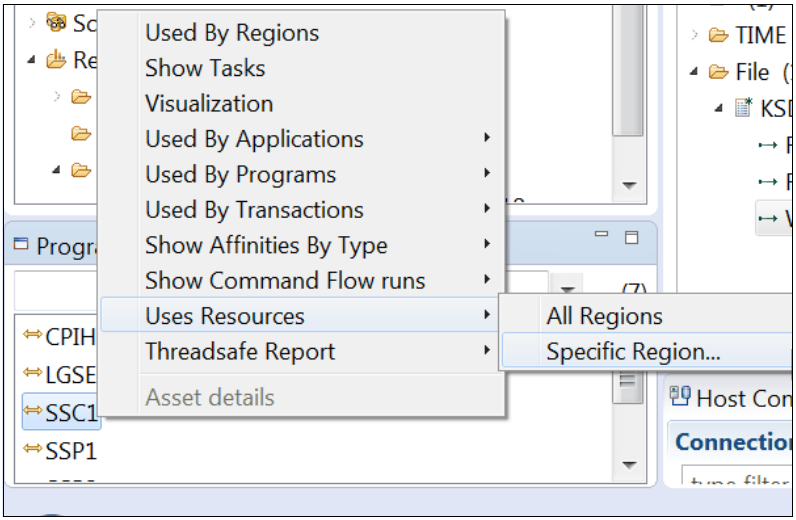


Figure 7-4 What resources are used by transaction SSC1

We select region IYDZZ518 and the Uses view displays the resources used by transaction SSC1, as shown in Figure 7-5 on page 187.

More information about the Uses view can be found in 3.1.2, “The Uses view” on page 67.

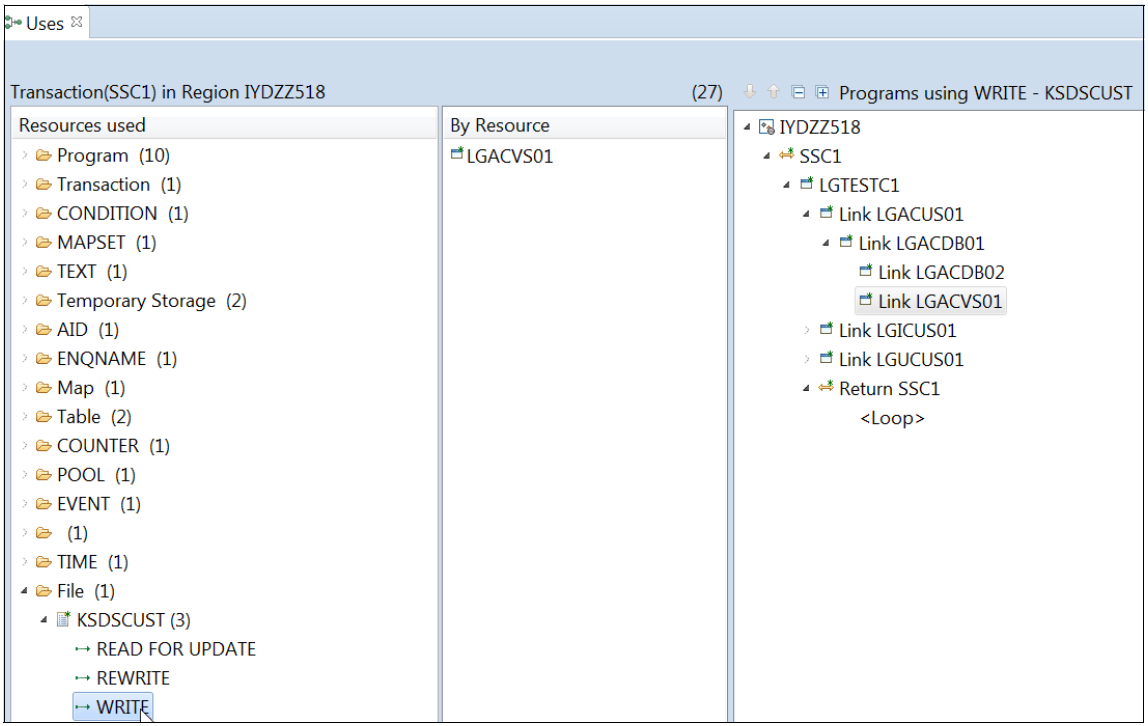


Figure 7-5 Resources used by transaction SSC1

We can see that the transaction uses file KSDSCUST and it issues an **EXEC CICS WRITE** against this file. We can assume from local knowledge that the WRITE is used to add a new customer record. By selecting the **WRITE** command, we can see that this command is issued by program LGACVS01.

Again, we can assume from local knowledge that program LGACVS01 adds a new customer record to the VSAM file KSDSCUST.

If you need to further understand the source code, you need to look at the source code. You can use IBM Rational Asset Analyzer with CICS IA to look at the actual source code. More information about Rational Asset Analyzer can be found in Appendix A, “IBM Rational Asset Analyzer” on page 225.

We drive the NEW_CUSTOMER event when program LGACVS01 is linked to. We only drive the event when the transaction is SSC1 and program LGACDB01 issues the link.

We can perform this action by right-clicking the LGACVS01 program in the By Resource section and selecting **Create Event → LINK PROGRAM** as shown in Figure 7-6.

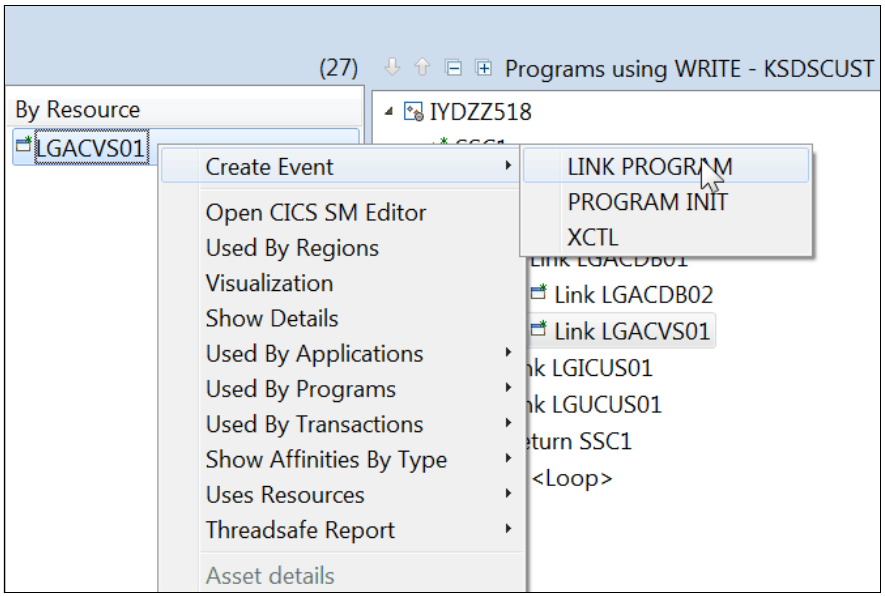


Figure 7-6 Creating a program link event

You then need to select a CICS region where this event will be deployed as shown in Figure 7-7.

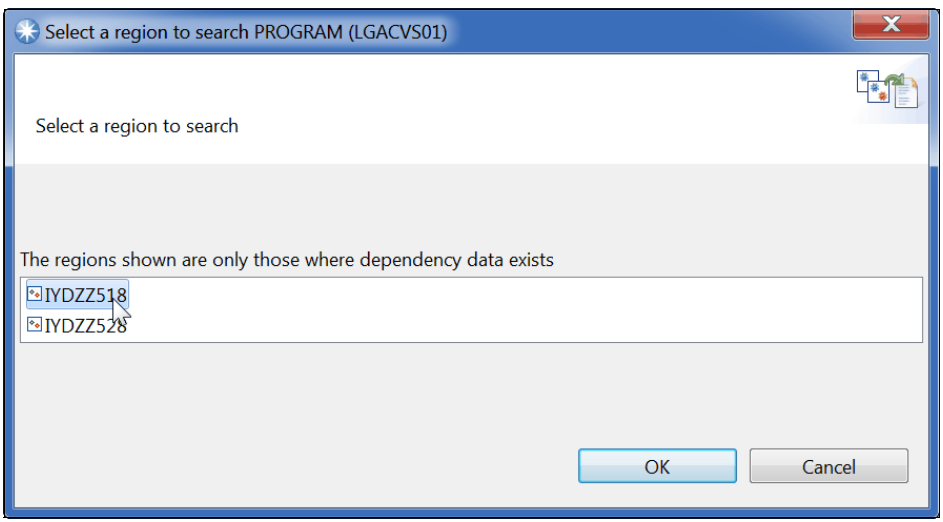


Figure 7-7 Select a CICS region

This action then drives the CICS TS Event wizard. First, you need to select the CICS Bundle project that you created previously and then give your event a name, as shown in Figure 7-8.

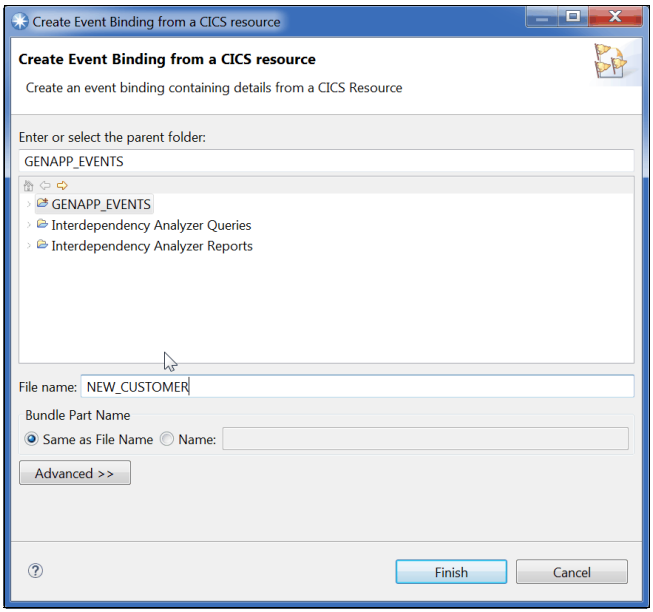


Figure 7-8 Create an event binding for your event

Click **Finish** and the Event Binding editor is opened as shown in Figure 7-9.

You can see that we created one event specification called NEW_CUSTOMER.

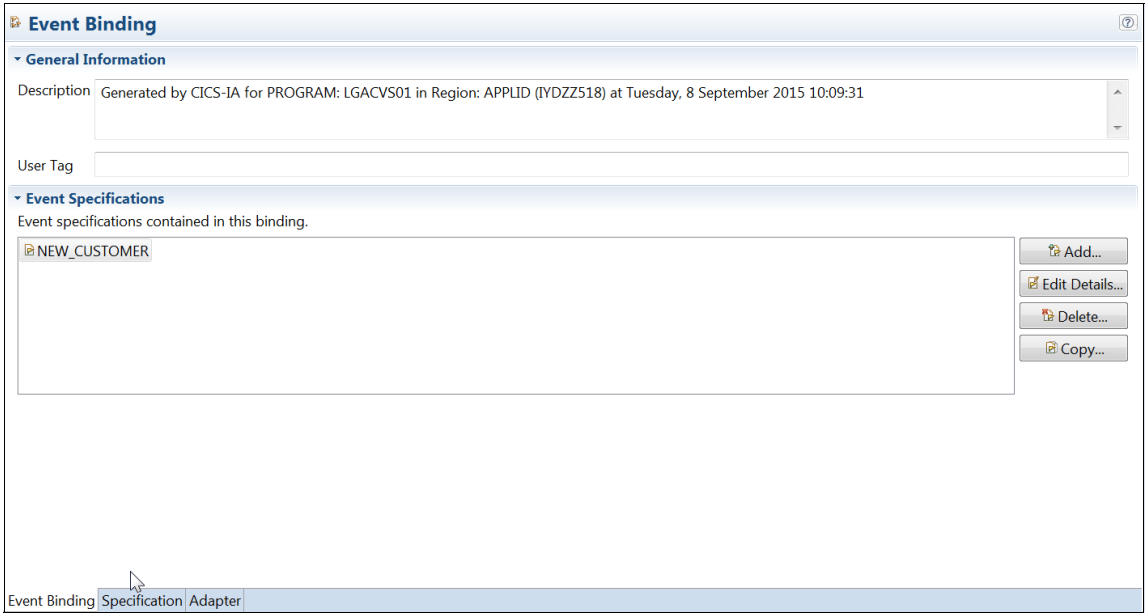


Figure 7-9 Event Binding editor

We can now click the `Specifications` folder shown in the lower-left corner and select the specification for “Capture_NEW_CUSTOMER” as shown in Figure 7-10. This figure shows the capture point. It has been pre-filled by CICS IA to select LINK PROGRAM as the capture point. If you want to emit an event and there is no suitable capture point in your application, for example, on a DB2 table update, you can use the CICS SIGNAL EVENT and the CICS IA to help in generating the event specification and filters.

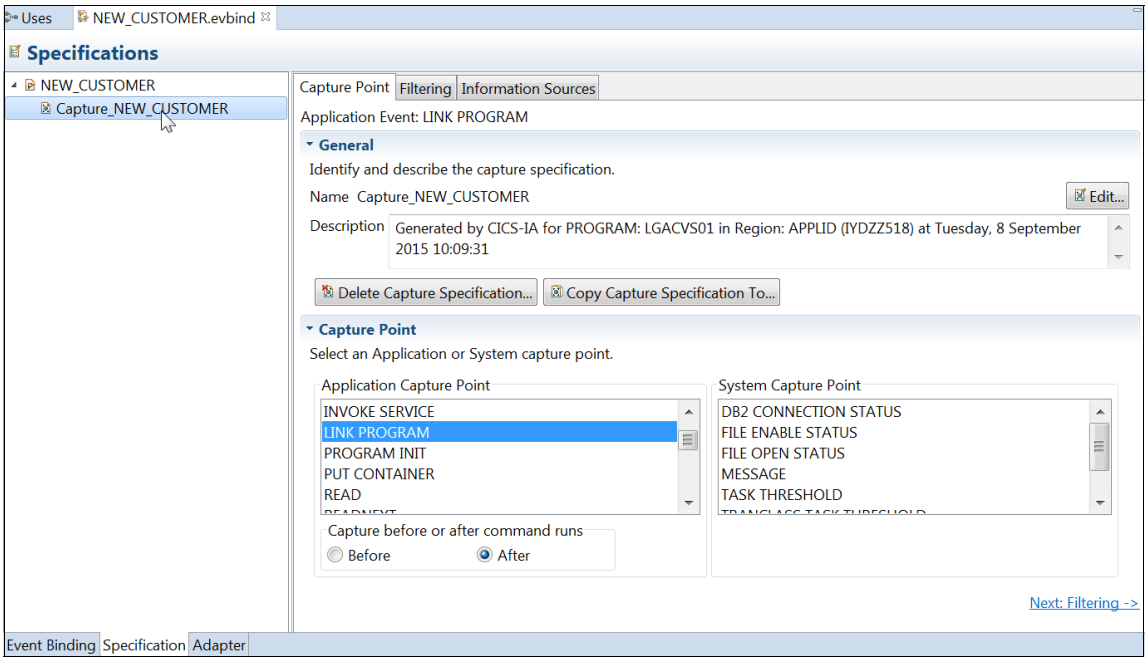


Figure 7-10 Event specification

We now look at the Filtering tab as shown in Figure 7-11.

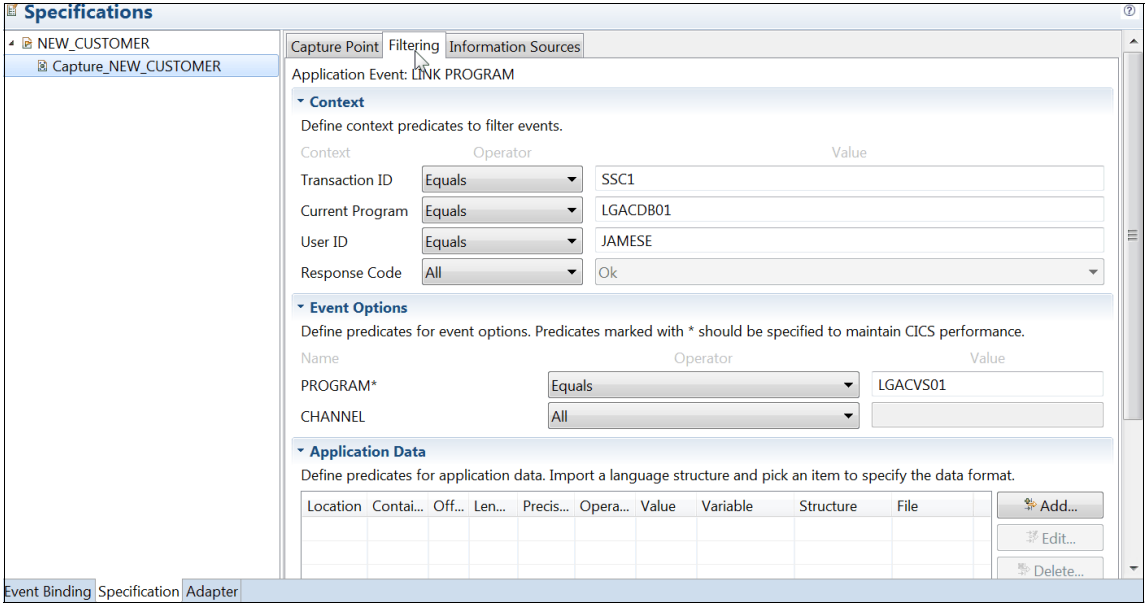


Figure 7-11 Event filtering

You can see that CICS IA pre-filled the following fields:

Transaction ID: SSC1
Current Program: LGACDB01
User ID: JAMESE

Under Event Options, we selected the linked to program as LGACVS01.

For testing purposes, you could leave the user ID set so the event is only emitted when running under your user ID.

You can then add further filtering by using Application Data, then select Information Resources, and finally select your adapter. Again, this process is described in “Chapter 6: Capture Application Events” in the Redbooks publication *Event Processing with CICS*, SG24-7792. See the following section for information about “Help from IBM” on page 240.

7.1.2 Capturing CICS IA Event data

To enable the collection of CICS Events data, you need to configure the collector to collect these events. To collect this data, you need to use the CINT transaction and select option 2 to “Configure Region Information”, then option 4 for “Options” against the default entry or a specific region entry, and then option 3 for “CICS API” calls. You can then change the CINT option for Events as shown in Figure 7-12.

```
CIU240        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/09/08
                        CICS Resources Options for                   10:41:04AM
          CICS Sysid   : DFTS     CICS Applid   : DEFAULTS

Modify the options and press Enter to update, or PF12 to Cancel.

Detect command types: Y=Yes, N=No
                      D=Yes+Detail ( Only for API types marked with * )
APIs
*Programs . . . D *Files. . . . . D *Transactions . D  Task Control . Y
 Presentation . Y *TS Queues  . . D *TD Queues  . . D  Journals . . . Y
 DTP  . . . . . Y  Counters . . . Y  FEPI . . . . . Y *WEB Services . D
*Exits  . . . . D  Others . . . . Y *EVENTS . . . . D  ATOMServices . Y
 XMLtransform . Y  WSAddressing . Y









CICS Sysid:  Z518   CICS Applid:  IYDZZ518   TermID:  TC15

F1=          F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Cancel
```

Figure 7-12 CICS API options

If you select “Y” against the Events option, we capture data when an event is emitted and when a CICS SIGNAL EVENT is issued.

If you select “D” for detailed information, as well as capturing the above, we also capture information about the Event Binding, Event Capture, and Adapter information.

We capture three resource types when an event is emitted:

- Event
- Event Capture Specification
- Event Binding

These types are shown when you request to see what resources are used by a transaction or program as shown in Figure 7-13. You also see that a SIGNAL EVENT has been issued.

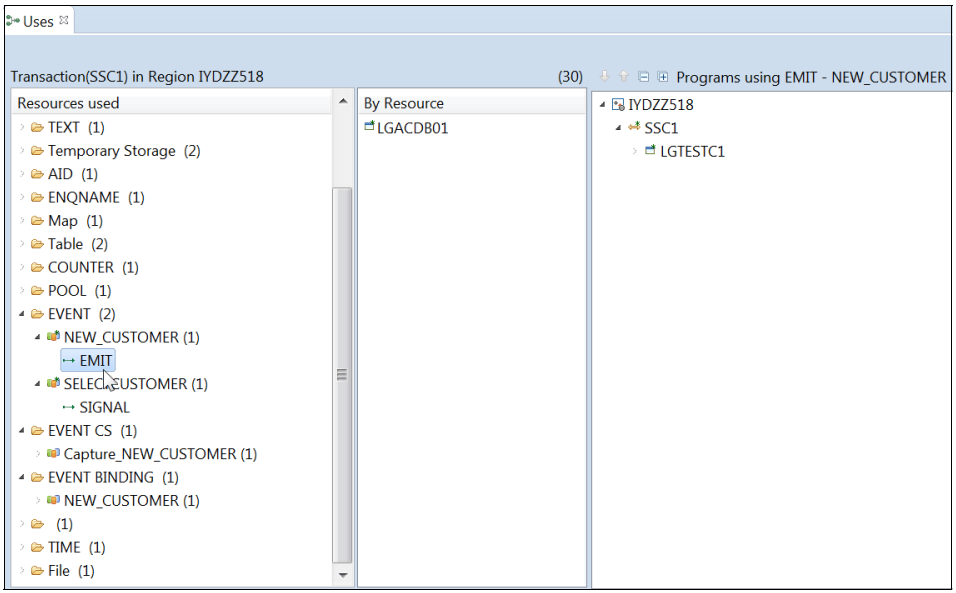


Figure 7-13 Event resources used by transaction SSC1

We also display the full event next to the command that was used for the capture point, in this case the PROGRAM LINK to program LGACVS01, as shown in Figure 7-14 on page 195.

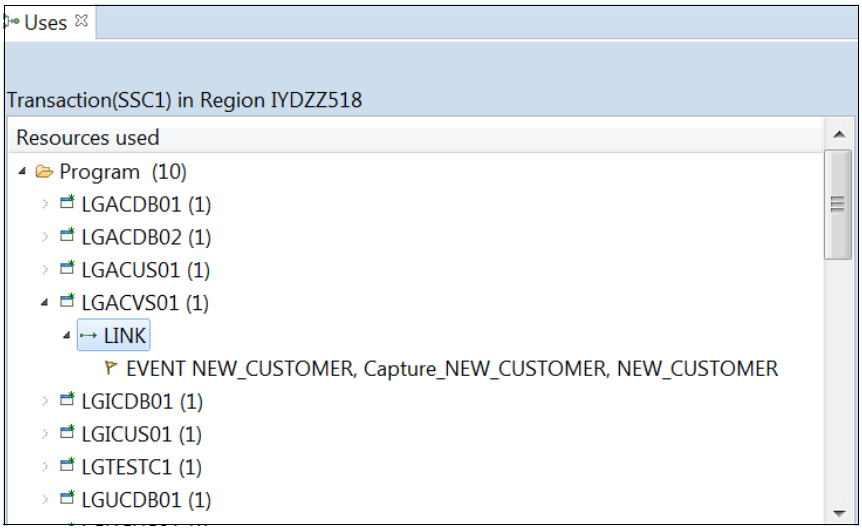


Figure 7-14 NEW_CUSTOMER event at capture point LINK PROGRAM

To open the Properties view, select **Window** → **Show View** → **Properties** from the menu bar (Figure 7-15).

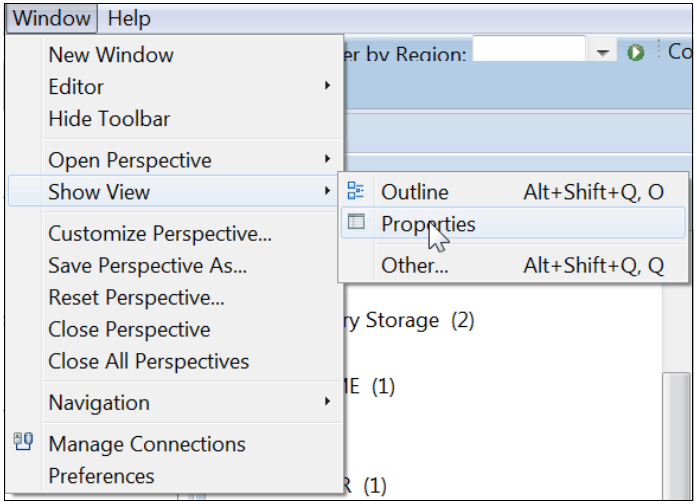


Figure 7-15 Opening the Properties view

Detailed event information is shown in the Properties view when you select the EVENT entry as shown in Figure 7-16.

Properties

Property	Value
Event NEW_CUSTOMER in Collection CICS510 in Region IYDZZ518, ,	
Application name	
Applid	IYDZZ518
Archive Date	
Base Scope of Bundle	
Bundle Creation Time	2015-09-08 11:05:46.0
Bundle Definition Source	TEST
Bundle Directory	/u/jamese/MailEvent/GENAPP_EVENTS_1.0.0/
Bundle Enabled Count	1
Bundle Installation Agent	GRPLIST
Bundle Installation Time	2015-09-08 11:09:46.0
Bundle Last Modification Agent	
Bundle Last Modification Agent Release	0680
Bundle Last Modification Time	2015-09-08 11:05:46.0
Bundle Last Modification User ID	JAMESE
Bundle Status	ENABLED
Capture Specification	Capture_NEW_CUSTOMER
Collection ID	CICS510
Event Binding	NEW_CUSTOMER
Event Binding Adapter	NEW_CUSTOMER
Event Binding Creation Time	2015-09-08 11:05:46.0
Event Binding Definition Source	NEWCUST
Event Binding Installation Agent	BUNDLE
Event Binding Installation Time	2015-09-08 11:09:46.0
Event Binding Installation User ID	CTOOLUSR
Event Binding Last Modification Agent	CSDAPI
Event Binding Last Modification Time	2015-09-08 11:05:46.0
Event Binding Last Modification User ID	JAMESE
Event Binding Status	ENABLED

Figure 7-16 EVENT DETAIL in the Properties view

8

Command Flow feature

In this chapter, we look at the CICS Interdependency Analyzer (IA) Command Flow feature. The Command Flow collector was added in CICS IA V3.1 in 2009. It was initially added to help with determining where TCB switches took place in a given task by capturing all CICS, DB2, IMS, and IBM MQ commands in chronological order. It was designed to be used by a single user, a systems programmer, to capture this data for a predefined number of transactions.

Since 2009, the collector has gone through many changes driven by customer requirements (RFEs) and now it can be used for the following purposes:

- Be used by multiple users including application developers
- Be used by a PRIVILEDGED user to capture data for all user IDs
- Capture the TCB switch count after each command
- Capture CPU response time between commands
- Capture transaction tracking data
- Use the transaction and program exclude lists by user

The CICS IA Explorer plug-in has also been enhanced to include:

- Visualization across regions
- Visualization across TCB switches
- Visualization across applications
- Search capability in Command Flow execution view
- Filtering in Command Flow execution view
- Command Flow administration and operation

We now look at how to perform the following functions:

- 8.1, “Administering the Command Flow collector” on page 198
- 8.2, “Loading the Command Flow data” on page 210
- 8.3, “Analyzing the Command Flow data” on page 212

8.1 Administering the Command Flow collector

The Command Flow collector options can be administered from three different places:

- The CINT transaction is used to add, remove, and edit the Command Flow user options.
- The CINC transaction is used to operate and administer individual Command Flow users.
- The CICS IA plug-in can also be used to operate and administer individual Command Flow users.

8.1.1 Adding, editing, and removing Command Flow users

To edit, add, or remove a Command Flow user, select option 4 for “User Administration” from the main CINT menu, as shown in Figure 8-1.

```
CIU000        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/16
                         Main Administration Menu                    11:12:33AM


Select one of the following.  Then press Enter.


4   1   Operations Menu.
    2   Configure Region Options.
    3   Configure Global Options.
    4   User Administration.









CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC77
CIU7000I 5655-Y22 (C) Copyright IBM Corp. 2001,2015
F1=Help      F2=           F3=Exit      F4=          F5=          F6=
F7=          F8=           F9=          F10=         F11=         F12=Exit
```

Figure 8-1 CINT user administration

From the User Administration Menu shown in Figure 8-2 you can add, copy, or delete a user. You can also edit the user details.

```
CIU400        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/16
                          User Administration Menu                   11:15:53AM

Type action code then press ENTER.                              Page : 1  of 1
1=Add User   2=Copy User   3=Delete User   4=User Details


 Act  USER ID   NEW USER  STATUS    JOURNAL   AUTHORITY
      AKHODAT             INACTIVE  CIUMTJNL  GENERAL
      ARISTO              INACTIVE  CIUMTJNL  GENERAL
      CICSUSER            INACTIVE  CIUMTJNL  GENERAL
      DOUGANV             INACTIVE  CIUMTJNL  GENERAL
      JAMESE              INACTIVE  CIUMTJNL  GENERAL
      VNDR69A             INACTIVE  CIUMTJNL  GENERAL









CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC77

F1=Help      F2=           F3=Exit      F4=          F5= Refresh  F6=
F7=Page Up   F8=Page Down  F9=          F10=         F11=         F12=
```

Figure 8-2 Add, copy, delete, edit users

You can use option 4 to edit the user options as shown in Figure 8-3.

```
CIU440        CICS Interdependency Analyzer for z/OS - V5R3M0        2015/10/16
                             User Details Menu                       11:19:05AM

  Options of CINC user JAMESE
  CINC Authority . . : GENERAL
  Traced transactions IDs. . . . : SSC1
  Traced user USERIDs  . . . . . : JAMESE
  Traced terminal TERMIDs  . . . : *
  Command flow data ID . . . . . : TEST
  Journal name for trace data. . : CIUMTJNL
  Dynamic call . . . . . : Y
  User modifiable exit name. . . :
  Traced regions APPLIDs . . . . : IYDZZ528


  Last data collection statistics
  Collector Status . . . . . : INACTIVE
  Collector last start . . . . . : 2015/08/21 09:41:51AM
  Collector last stop. . . . . : 2015/08/21 09:41:37AM

CICS Sysid:  Z538   CICS Applid:  IYDZZ538   TermID:  TC77

F1=Help      F2=           F3=Save&Exit F4=          F5= Refresh  F6=
F7=          F8=           F9=          F10=         F11=         F12= Cancel
```

Figure 8-3 Edit the user options

As the administrator, you can change the following options:

- CINC Authority
- Journal name for trace data

CINC Authority

The CINC Authority option is used to control what options are available to the Command Flow user. It has two values:

GENERAL: This value is the default and it allows the Command Flow user to run the collector to capture data for *one* user ID only. The user ID can be the collector user ID or any other user ID.

PRIVILEDGED: This value allows the Command Flow user to capture data for all user IDs or subset of user IDs defined by a wildcard. This option is equivalent to running the original command flow collector as a single user.

Note: Only one PRIVILEDGED collection can be run at one given time. To run a PRIVILEDGED collection for more than one user ID, there can be no other Command Flow runs active.

Journal name for trace data

This option allows you to define individual user journals for a Command Flow user. By default, all the Command Flow data is written to one default logstream as defined by the Journal Mode definition shipped with CICS IA. The default value for this option is CIUMTJNL.

8.1.2 Using CINC to collect Command Flow data

The CICS CINC transaction is used by individual Command Flow users to administer and operate the collection. You can use the various options to control what data you collect.

The following options are available:

- Command Flow ID
- Traced user USERIDs
- Traced terminal TERMIDs
- Traced transaction IDs
- Exclude lists
- Journal Copy Criteria
- User Exit Name
- Dynamic Call

- Tasks before stopping
- Records before stopping

You can use the PFKEYS, F4 (to start) and F6 (to stop) the collector.

```
CIUAO1            CICS IA Command Flow Options                ApplID IYDZZ538


Command Flow State.  .  .  . : STOPPED
Date/Time of last start .  . : 2015/08/21   09:41:51AM
Date/Time of last stop  .  . : 2015/08/21   09:41:37AM

Command Flow ID.  .  .  .  . : TEST

Traced user USERIDs  .  .  . : JAMESE            CINC Authority : GENERAL
Traced terminal TERMIDs .  . : *

Traced transaction IDs  .  . : SSC1                     (Max 5 transact. IDs)
Exclude lists  .  .  .  .  . : Transaction              Program
User Journal Name .  .  .  . : CIUMTJNL
Journal Copy Criteria.  .  . : LAST       (LAST, USER or CFID)
User Exit Name .  .  .  .  . :
Dynamic Call.  .  .  .  .  . : Y          (Y=Yes, N=No)
Tasks before stopping   .  . : 0          (0-9999)
Records before stopping .  . : 0          (0-9999999)


F1=Help  F2=         F3=Exit     F4=Options       F5=Start          F6=Stop
F7=Stats F8=         F9=         F10=Applications F11=              F12=Cancel
```

Figure 8-4 CINC Command Flow options

Command Flow ID

This option is the name associated with the Command Flow collection. It can be up to eight characters.

Traced user USERIDs

This option is the CICS user ID for which we capture the data. It is not to be confused with the user ID invoking the collection. In most cases, when the Command Flow is used by application developers, the invoking user ID and the traced user ID will be one in the same. To use generic user IDs, the invoking user must be a PRIVILEDGED user ID. For more information about PRIVILEDGED users, see section “CINC Authority” on page 200.

Traced terminal TERMIDs

You can use this option to capture data for a specific terminal or generic terminals. To capture all terminals, enter an asterisk (*).

Traced transaction IDs

You can use these fields to select for which transactions you want to capture data. You can enter up to four transactions or generic transactions. Again, for all transactions you can use an asterisk (*).

Exclude lists

You can use these fields to exclude certain transaction or programs from the data captured. For more information about using exclude lists, see 2.1.3, “Using the IA exclude lists” on page 29.

Journal Copy Criteria

This option is used to define what data is offloaded from the logstream into the GDG data set for each Command Flow user. It has three possible values:

- **LAST**: Copy the last collection for the Command Flow user.
- **USER**: Copy all the records for the Command Flow user.
- **CFID**: Copy only the records based on the Command Flow trace ID.

User Exit Name

This option allows you to enter an eight-character user-modifiable exit name that you can use to add data to a user’s Command Flow records. For more information, see section “The CICS IA Command Flow user exit” in the CICS IA Users Guide, which is in the IBM Knowledge Center:

`http://www.ibm.com/support/knowledgecenter/SSPPUS/welcome.html`

Dynamic Call

This option allows you to switch on and off the dynamic call option. For more information, see 2.1.5, “Dynamic COBOL calls” on page 39.

The final two options allow you to stop the collector under certain circumstances. This capability was added to allow you to run the Command Flow in a production environment to assist with problem resolution for a given transaction.

Tasks before stopping

The number of tasks executed before stopping the collector. A value of 0 switches the feature off.

Records before stopping

The number of records written to the logstream before stopping the collector. A value of 0 switches the feature off.

You can collect Command Flow data across CICS regions. The CICS regions must be defined in the CIUCNTL file using the CINT transaction. They need to have an active connection and to get the best value they must share the logstream used to capture the Command Flow data. Use the PFKEY F5 for “Options” to add or remove CICS regions as shown in Figure 8-5.

```
CIUAO2            CICS IA Command Flow ApplID list            ApplID IYDZZ538

Press F4 for select the region OR type regions and press ENTER

CICS ApplIDs.  .  .  .  .  : IYDZZ528
                           :
                           :

F1=Help    F2=    F3=Exit     F4=Prompt for regions    F5=Delete    F6=
F7=        F8=    F9=         F10=             F11=                F12=Cancel
```

Figure 8-5 Command Flow ApplID list

Use PFKEY F5 to delete an applid.

Use PFKEY F4 to prompt for available regions as shown in Figure 8-6.

```
CIUAO4              List of available CICS regions            ApplID IYDZZ538

Select the required region and press F3.                        More:


      CICS         CICS         CICS         CICS         CICS         CICS
 s   ApplID   s   ApplID   s   ApplID   s   ApplID   s   ApplID   s   ApplID

    IYDZZ538     IYDZZ518     IYDZZ528
```

Figure 8-6 Enter s to select an ApplID

You can also choose to collect CICS TS platform and application data associated with a task when running a Command Flow collection. Use PFKEY F10 to switch on application collection as shown in Figure 8-7.

```
CIUAOA     CICS IA Command Flow Application Data Collection    ApplID IYDZZ538


Type Collection Option then press ENTER :

Enable collection of Application Data: N

Selected Application:
ALL.
```

Figure 8-7 Switch on collection of application data

8.1.3 Using the IA plug-in to collect Command Flow data

In the previous section, we looked how the CINC transaction can be used to administer and operate the Command Flow collector. In this section, we look at how we can achieve the same by using the CICS IA plug-in. We configure the collector to collect data for the **SSC1** transaction for the default user ID **CICSUSER**. The collection is performed under the user ID **JAMESE**.

Note: It is important to distinguish between the two user IDs associated with the Command Flow collection. The following two user IDs are used:

- Collector or Owner User ID: The user ID running the collection. In this case, **JAMESE**.
- Collected User ID: The user ID for which we are collecting data. In this case, **CICSUSER.**

The options that are available in the IA plug-in are the same as those options available via the CINC transaction. The options are described in 8.1.2, “Using CINC to collect Command Flow data” on page 200.

The CICS IA plug-in provides the capability to operate and administer the dependency, affinity, and Command Flow collector via a CICS Webservice. Before you can operate the Command Flow collector, you need to log on to the Webservice by using the **IA Collection Connection**.

Logging on to the IA Collector Connection

The IA Collector Connections are only selectable when you are in the IA Operations view as shown in Figure 8-8 on page 205.

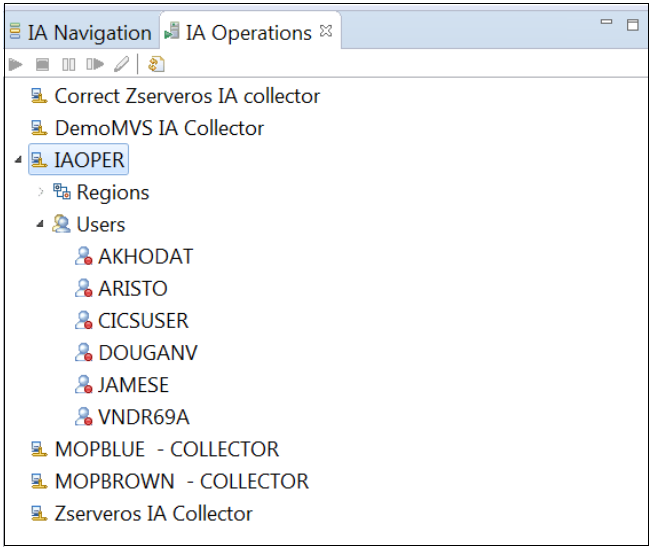


Figure 8-8 IA Operations view

All Explorer connections are available via the Host Connections view. The simplest way to access this view and manage connections is to select the down arrow icon as shown in Figure 8-9.

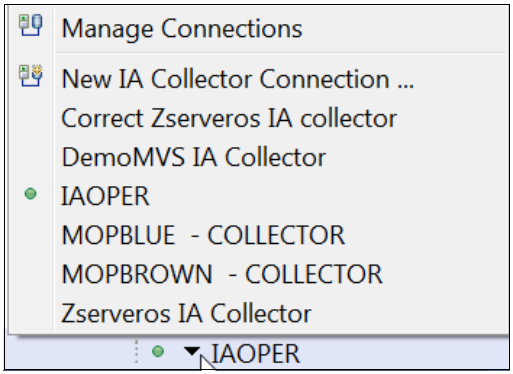


Figure 8-9 Manage connections

From here, you can do the following actions:

- Select **Manage Connections** to open the Host Connections view
- Add a **New IA Collector Connection**
- Select an existing **IA Collector Connection**

In this case, we select an existing connection called IAOPER.

When adding a *new* connection, you need to configure the following options:

- **Name**: The local name that you give to this database connection. Use this name to distinguish between different database connections.
- **Host name**: The host name or IP address of the z/OS host system for the interface to the CICS regions for which you want to collect data.
- **Port number:** The port number for the interface to the CICS regions for which you want to collect data.
- **Secure connection**: Select this check box if the connection uses SSL security.

These options are shown in Figure 8-10.

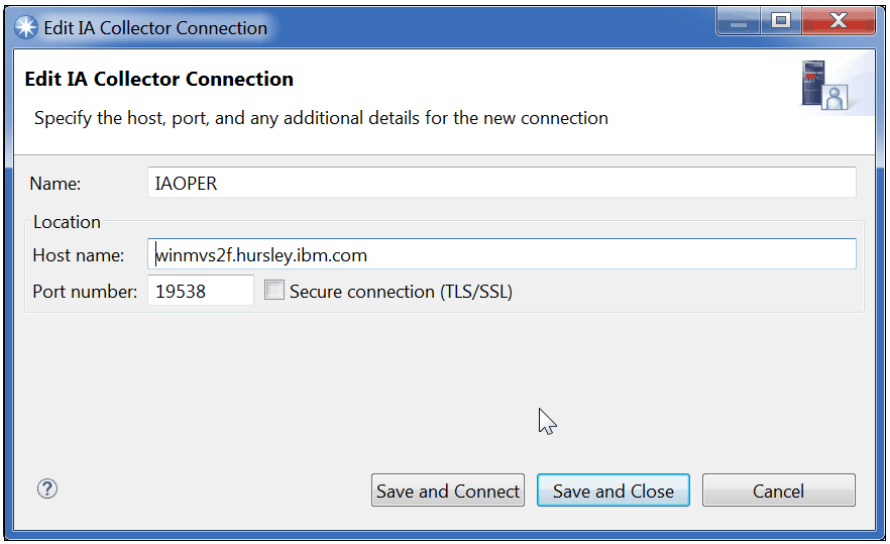


Figure 8-10 IA Collector options

You can now configure the collector options to collect data for transaction SSC1.

Configuring the IA Command Flow collector options

To configure the IA Command Flow option, right-click your user ID in the IA Operations view, and choose **Edit collector options** as shown in Figure 8-11.

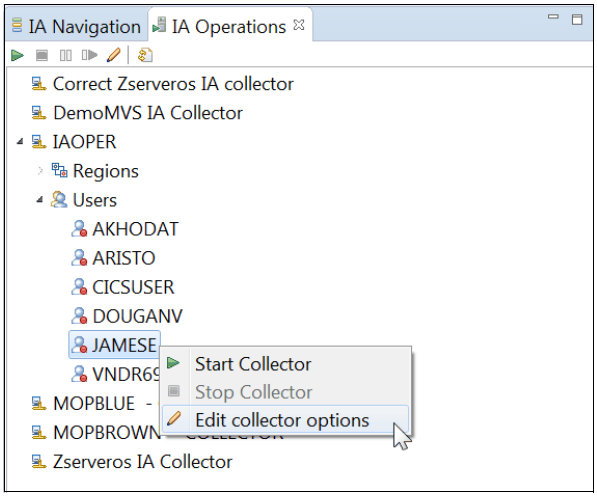


Figure 8-11 Edit collector options for user JAMESE

This action opens an editor view where you can configure your options as shown in Figure 8-12.

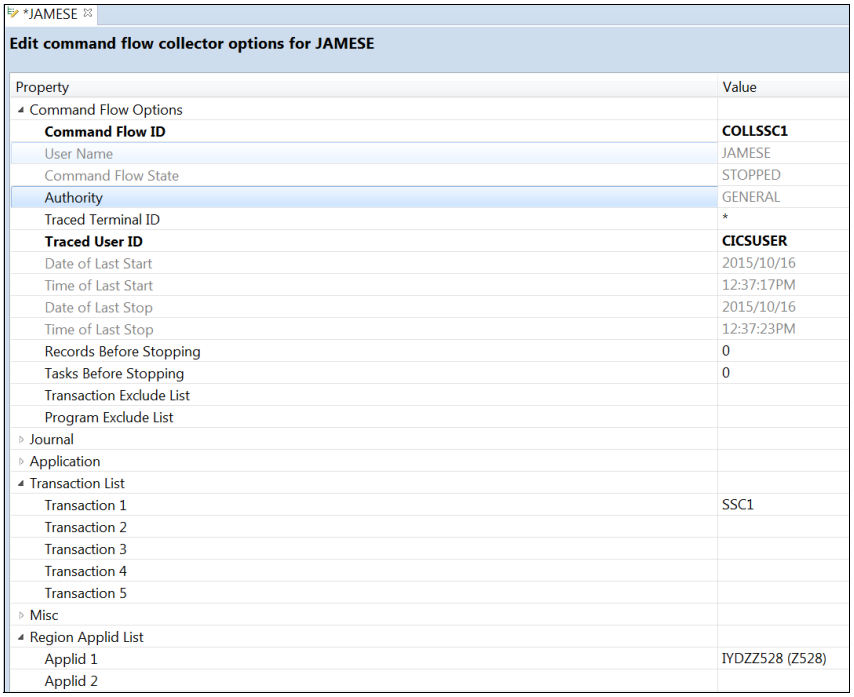


Figure 8-12 Set the Command Flow collector options

In this example, we set the following options:

- **Command Flow ID**: COLLSCC1
- **Trace User ID**: CICSUSER
- **Transaction 1**: SSC1
- **Applid 1**: IYDZZ528
- **Journal Copy Criteria**: CFID

After you select your options, you need to save the options by using **CTRL+S** or by using **File** → **Save** from the upper-right corner.

You can now start the collector. Right-click your user ID and select **Start Collector** as shown in Figure 8-13.

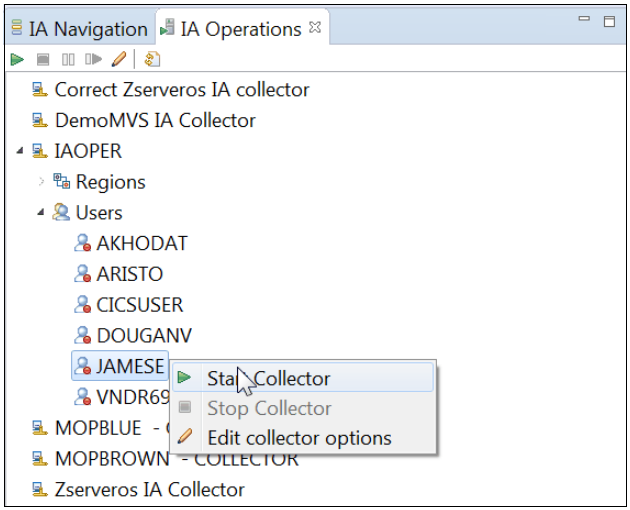


Figure 8-13 Start the Command Flow collector

The status of the collector can be seen in the Properties view as shown in Figure 8-14. To open the Properties view, select **Window → Show View → Properties**.

Properties

Property	Value
◢ Applid 1	
Applid	IYDZZ528
Collector/Region State	RUNNING
Records Written	0
Sysid	Z528
◢ User Details	
Application Filter	N
Application Name	
Authority	GENERAL
Command Flow ID	COLLSSC1
Date of Last Start	2015/10/18
Date of Last Stop	2015/10/18
Dynamic Call	Y
Journal Copy Criteria	CFID
Journal Name	CIUMTJNL
Major Version	0
Micro Version	0
Minor Version	0
Program Exclude List	
Records Before Stopping	0
State	RUNNING
Tasks Before Stopping	0
Time of Last Start	11:54:02AM
Time of Last Stop	11:19:29AM
Traced Terminal ID	*
Traced User ID	CICSUSER
Transaction 1	SSC1

Figure 8-14 Command Flow collector options

You can now run the SSC1 transaction and then stop the Command Flow collection. To stop the collector, right-click your user ID and select **Stop Collector**. The Properties view is updated with the number of records collected. You can now load the Command Flow data.

8.2 Loading the Command Flow data

There are three steps to loading the data into the CICS IA DB2 tables:

1. CIUJLCPY: Extract logstream records to a GDG data set
2. CIUUPDB5: Load the data into DB2 from the GDG data set
3. CIUJLDEL: Delete the loaded data from the logstream

The sample job CIUJLCPY extracts records from the logstream based on the Command Flow collector options for the collecting user ID, in our case **JAMESE**. The sample JCL for CIUJLCPY is shown in Figure 8-15.

```
//*-------------------------------------------------------------
//*       COPY RECORDS FROM LOG STREAM DATA SETS TO GDG QSAM FILE
//*-------------------------------------------------------------
//    SET  USERID=&SYSUID                                      @M6A
//STEP010  EXEC PGM=CIUU044,PARM='&USERID'                      @M6C
//STEPLIB  DD   DSN=ANTZ.CICS.IA.DEV.BSF.SCIULOAD,
//              DISP=SHR
//         DD   DSN=ANTZ.CICS.IA.DEV.BSF.SCIULODE,
//              DISP=SHR
//SYSOUT   DD   SYSOUT=*
//CIUPRINT DD   SYSOUT=*
//CIUCNTL  DD   DSN=CICSIA53.V10.CIUCNTL,DISP=SHR               @M8A
//JNLINPUT DD   DSNAME=CICSIA53.V10.CIUMTJNL,
//              DCB=(BLKSIZE=32760),
//              SUBSYS=(LOGR,DFHLGCNV)
//CIUCMDTR DD   DSNAME=CICSIA53.V10.&USERID..CIUCMDFL(+1),      @M6C
//              DISP=(NEW,CATLG),
//*             DATACLAS=_smsdatac_,
//*             STORCLAS=_smsstorc_,
//*             MGMTCLAS=_smsmngc_,
//              UNIT=SYSDA,SPACE=(CYL,(5,5),RLSE),
//              DCB=(RECFM=FB,LRECL=807,BLKSIZE=31473)       @R100506C
```

Figure 8-15 JCL for the CIUJLCPY job

You see that the user ID running the extract job is passed as a parameter in to program CIUU040. This program reads the user record from the CIUCNTL file and obtains the Journal Copy Criteria to be used to select the records to be extracted. If the user ID running the extract job is different from the collector user ID, use the // SET USERID= statement to set the user ID to the collector user ID.

In our case, the collector user ID and the user ID submitting the job is JAMESE. The Journal Copy Criteria used for this collection is CFID as shown in Figure 8-14 on page 210. This option implies that we extract all the data with a collection ID that matches the selected collection ID, in our case COLLSSC1 for user ID JAMESE. The other options for Journal Criteria Copy options are described in “Journal Copy Criteria” on page 202.

The sample job CIUJLCPY should complete with RC=0 and the output file CIUPRINT should include extract details as shown in Figure 8-16.

```
CIU6019I  Journal records read  = 000006617

CIU6020I  Trace records written = 000000275
```

Figure 8-16 Extract report for job CIUJLCPY

After running job CIUJLCPY, you need to run job CIUUPDB5 to load the database. This job takes as input the GDG data set that was updated in the previous job. Again, you can use the // SET USERID= statement to set the user ID to the collector user ID.

You should now have Command Flow data available for analysis.

8.3 Analyzing the Command Flow data

The Command Flow collections that we collected are available from the User Command Flow branch in the IA Navigation view as shown in Figure 8-17. They are listed by Owner User ID (Collector User ID), followed by Collection ID name, followed by the collected user ID, followed by the time stamp and the task ID.

You use the data collected in the previous section. You can see from Figure 8-17 that the owner user ID is JAMESE, the collection ID is COLLSSC1, and the collected user ID is CICSUSER.

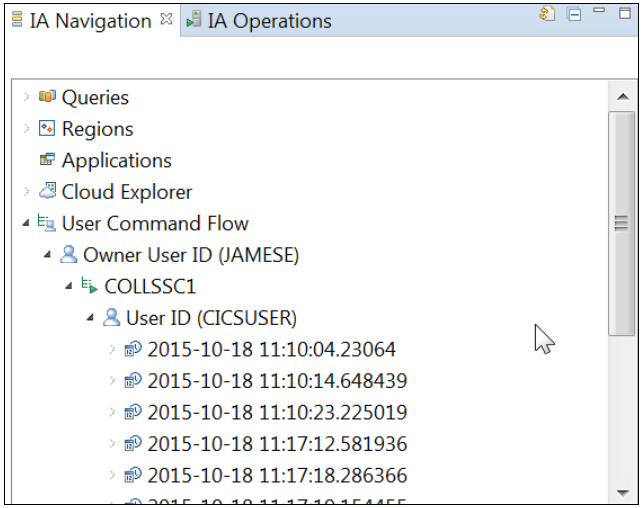


Figure 8-17 User command flows

There are two methods of viewing the command flow data for a task:

- The Execution view
- The Visualization views

8.3.1 The Command Flow Execution view

Right-click the required task and select **Show Execution** as shown in Figure 8-18.

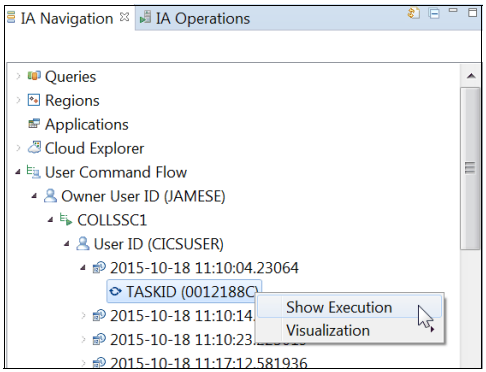


Figure 8-18 Show Execution option

This action opens the Command Flow view as shown in Figure 8-19.

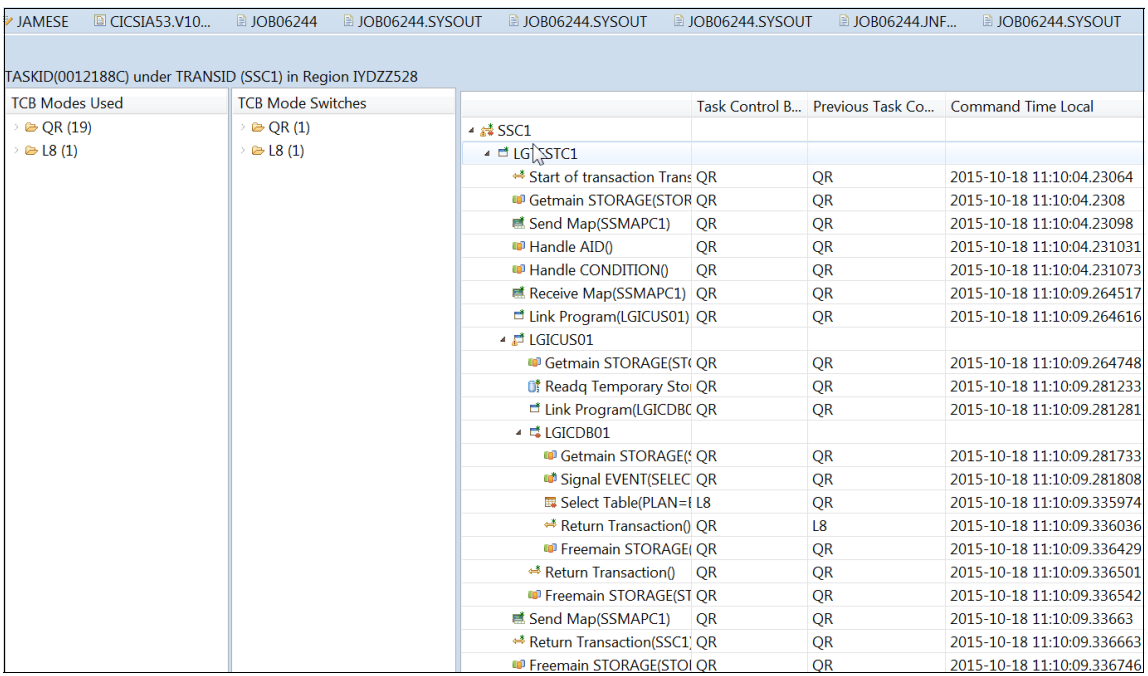


Figure 8-19 Command Flow execution view

This view is historically broken down into three parts:

- TCB Modes Used: The command execution count by TCB mode
- TCB Mode Switches: The command count causing a TCB mode switch
- The execution tree: The command execution tree

The Command Flow collector was originally designed to assist with CICS threadsafe analysis and as you can see the default view is designed as such. In this section, we focus on the execution part of the view and how we can modify the view for use by application developers. The threadsafe aspects of this view are covered in 5.4, “Using the command flow data to analyze TCB swaps” on page 135.

The default execution view lists the commands used by the transaction, then the initial program, then the commands. Indentation is used to show when one program LINKs, XCTLs, or CALLs another program. These features are shown in Figure 8-20.

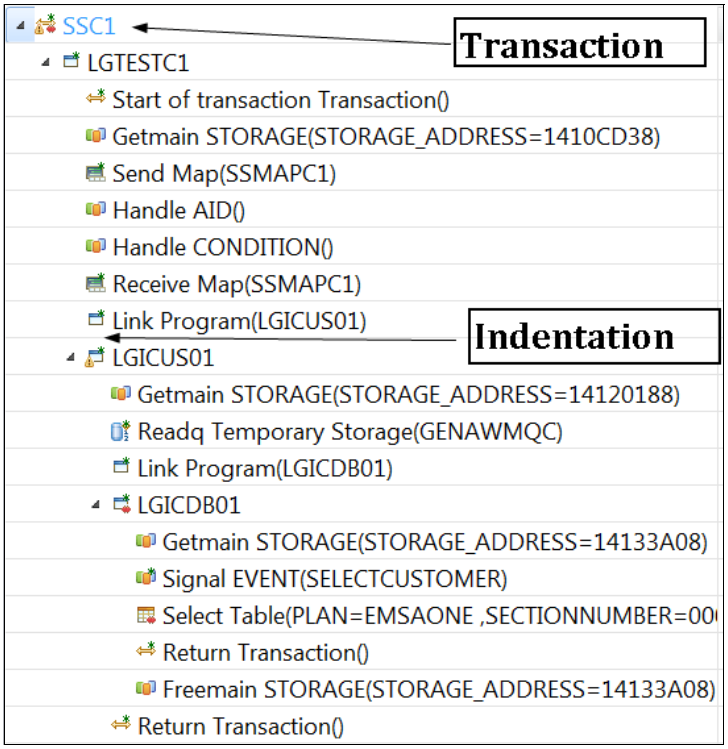


Figure 8-20 Command Flow execution

We now look at other features that are available in the Command Flow view:

- Customize the columns shown in the view

- Use the filter options
- Use the **FIND** command

Customizing columns

To customize the columns, select the **Column Menu** in the upper-right menu bar and select **Customize columns** as shown in Figure 8-21.

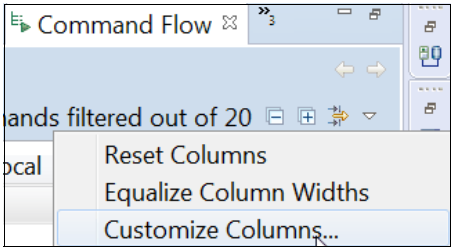


Figure 8-21 Customize columns

We remove the TCB columns as shown in Figure 8-22.

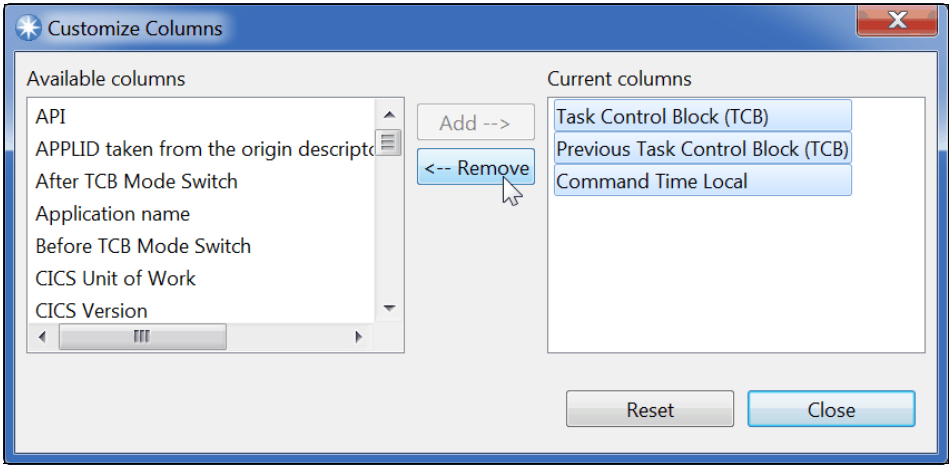


Figure 8-22 Remove TCB columns

You then add the following columns as shown in Figure 8-23 on page 216:

- Command
- Resource type
- Resource name
- Response Code
- Reason Code

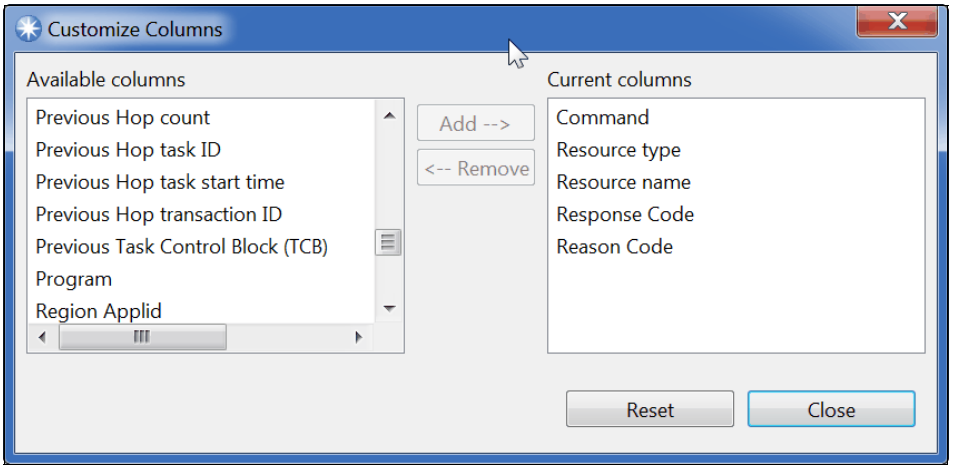


Figure 8-23 Add new columns

Our execution view now contains data that could be used by an application developer as shown in Figure 8-24.

SSC1					
LGTESTC1					
Start of transaction Transaction()	START OF TRANSACTION	TRANSID			
Getmain STORAGE(STORAGE_ADDRESS=1	GETMAIN	STORAGE	STORAGE_ADDRESS=1410CD38	00000000	00000000
Send Map(SSMAPC1)	SEND	MAP	SSMAPC1	00000000	00000000
Handle AID()	HANDLE	AID		00000000	00000000
Handle CONDITION()	HANDLE	CONDITION		00000000	00000000
Receive Map(SSMAPC1)	RECEIVE	MAP	SSMAPC1	00000000	00000000
Link Program(LGICUS01)	LINK	PROGRAM	LGICUS01	00000000	00000000
LGICUS01					
Getmain STORAGE(STORAGE_ADDRESS	GETMAIN	STORAGE	STORAGE_ADDRESS=14120188	00000000	00000000
Readq Temporary Storage(GENAWMQC	READQ	TSQUEUE	GENAWMQC	0000002C	00000000
Link Program(LGICDB01)	LINK	PROGRAM	LGICDB01	00000000	00000000
LGICDB01					
Getmain STORAGE(STORAGE_ADDRE	GETMAIN	STORAGE	STORAGE_ADDRESS=14133A08	00000000	00000000
Signal EVENT(SELECTCUSTOMER)	SIGNAL	EVENT	SELECTCUSTOMER	00000000	00000000
Select Table(PLAN=EMSAONE ,SECTI	SELECT	TABLE	PLAN=EMSAONE ,SECTIONNUMBER=00...		
Return Transaction()	RETURN	TRANSID		00000000	00000000
Freemain STORAGE(STORAGE_ADDR	FREEMAIN	STORAGE	STORAGE_ADDRESS=14133A08	00000000	00000000
Return Transaction()	RETURN	TRANSID		00000000	00000000
Freemain STORAGE(STORAGE_ADDRES	FREEMAIN	STORAGE	STORAGE_ADDRESS=14120188	00000000	00000000
Send Map(SSMAPC1)	SEND	MAP	SSMAPC1	00000000	00000000
Return Transaction(SSC1)	RETURN	TRANSID	SSC1	00000000	00000000
Freemain STORAGE(STORAGE_ADDRESS=	FREEMAIN	STORAGE	STORAGE_ADDRESS=1410CD38	00000000	00000000

Figure 8-24 An application developer's view of the Command Flow execution

This view can be saved for future use by saving the perspective using **Window → Save Perspective As** as shown in Figure 8-25.

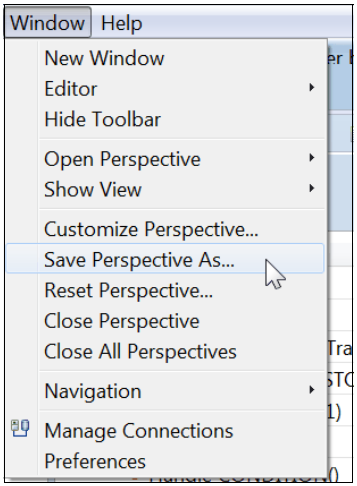


Figure 8-25 Save perspective

You can then replace your existing IA perspective or create a new one as shown in Figure 8-26 on page 218.

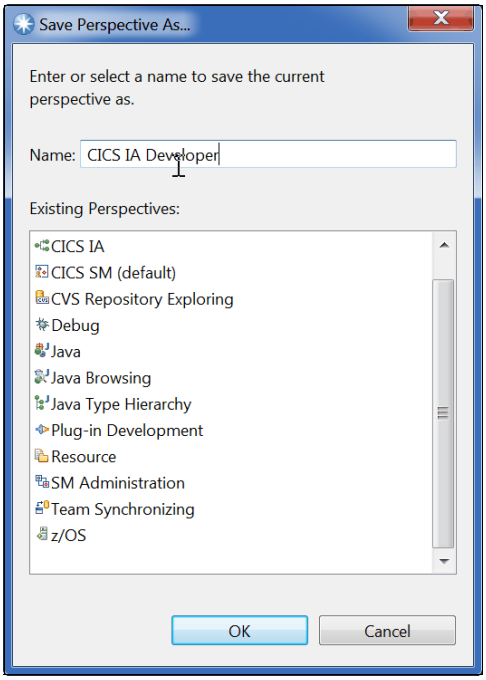


Figure 8-26 Create a CICS IA Developer perspective

Filtering data

To filter the data shown in the execution view, select the **Show Filters** icon in the upper-right menu bar as shown Figure 8-27.

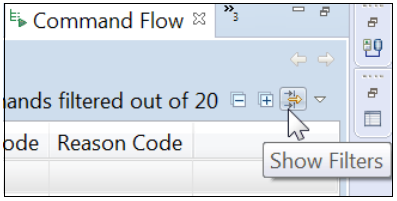


Figure 8-27 Show Filters icon

The filter list appears on the right side of the view and you can filter on the resource type or the command, as shown in Figure 8-28.

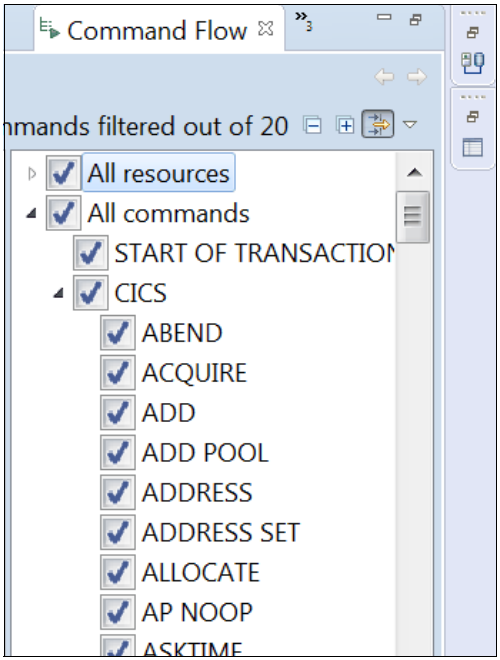


Figure 8-28 Filtering options

You now use the filtering to show only PROGRAM resources. First, clear the “All resources” box shown in Figure 8-28. Then, select the **PROGRAM** resource as shown in Figure 8-29.

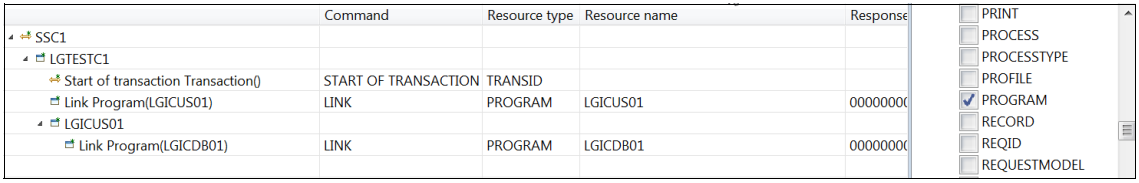


Figure 8-29 PROGRAM flow in the execution tree

This view gives you a display of the program flow within the execution tree as shown in Figure 8-29.

Using the Find feature

To demonstrate this feature, you collapse the Execution view as shown in Figure 8-30.

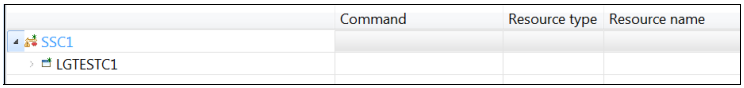


Figure 8-30 Collapsed Execution view

You then enter **CTRL+F** from within the Execution view. This action opens the Find window that is shown in Figure 8-31.

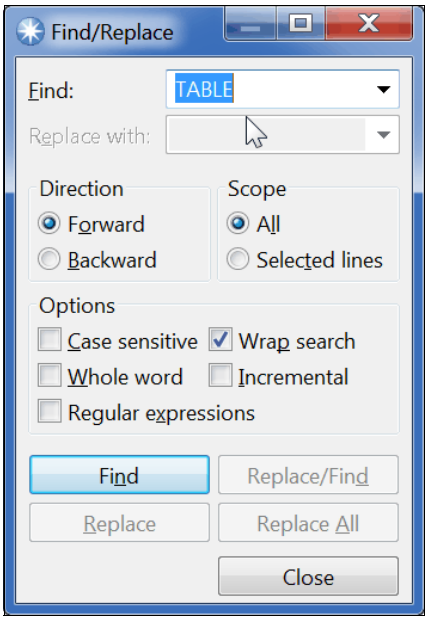
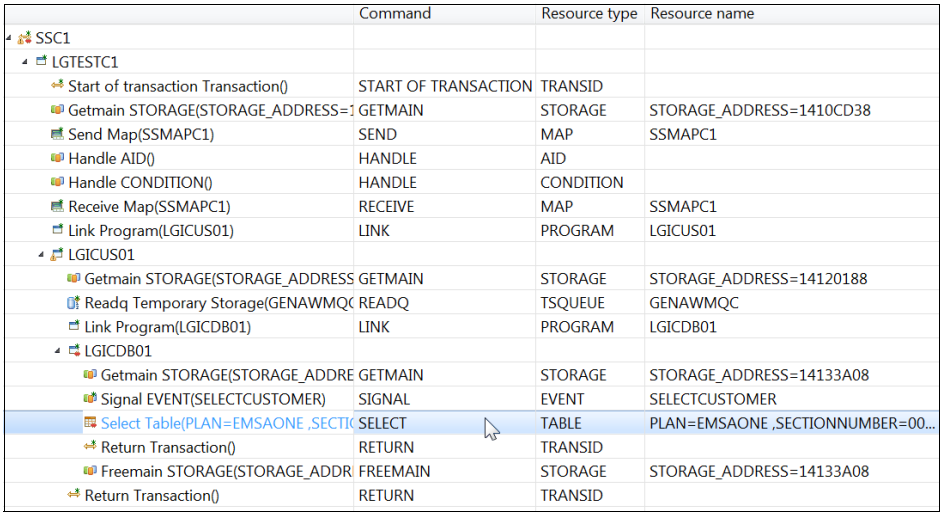


Figure 8-31 Find window

You can search on resource type, resource name, and so on. In this example, we search for a DB2 TABLE. Pressing **Find** causes the Execution view to expand and highlight the command containing a DB2 table, as shown in Figure 8-32 on page 221.



	Command	Resource type	Resource name
SSC1			
LGTESTC1			
Start of transaction Transaction()	START OF TRANSACTION	TRANSID	
Getmain STORAGE(STORAGE_ADDRESS=1	GETMAIN	STORAGE	STORAGE_ADDRESS=1410CD38
Send Map(SSMAPC1)	SEND	MAP	SSMAPC1
Handle AID()	HANDLE	AID	
Handle CONDITION()	HANDLE	CONDITION	
Receive Map(SSMAPC1)	RECEIVE	MAP	SSMAPC1
Link Program(LGICUS01)	LINK	PROGRAM	LGICUS01
LGICUS01			
Getmain STORAGE(STORAGE_ADDRESS	GETMAIN	STORAGE	STORAGE_ADDRESS=14120188
Readq Temporary Storage(GENAWMQ(	READQ	TSQUEUE	GENAWMQC
Link Program(LGICDB01)	LINK	PROGRAM	LGICDB01
LGICDB01			
Getmain STORAGE(STORAGE_ADDRE	GETMAIN	STORAGE	STORAGE_ADDRESS=14133A08
Signal EVENT(SELECTCUSTOMER)	SIGNAL	EVENT	SELECTCUSTOMER
Select Table(PLAN=EMSAONE ,SECTI	SELECT	TABLE	PLAN=EMSAONE ,SECTIONNUMBER=00...
Return Transaction()	RETURN	TRANSID	
Freemain STORAGE(STORAGE_ADDR	FREEMAIN	STORAGE	STORAGE_ADDRESS=14133A08
Return Transaction()	RETURN	TRANSID	

Figure 8-32 FIND a command containing a DB2 TABLE

8.3.2 Command Flow visualization

You can choose to analyze your Command Flow data using new graphical visualization added in CICS IA V5.2. We provide three different types of visualization:

- Application switches: Shows the execution across CICS applications
- Region switches: Shows the execution across CICS regions
- TCB switches: Shows the execution across TCB modes

In this example, we show the execution across TCB switches.

Right-click the required task and select **Visualization → TCB Switches** as shown in Figure 8-33.

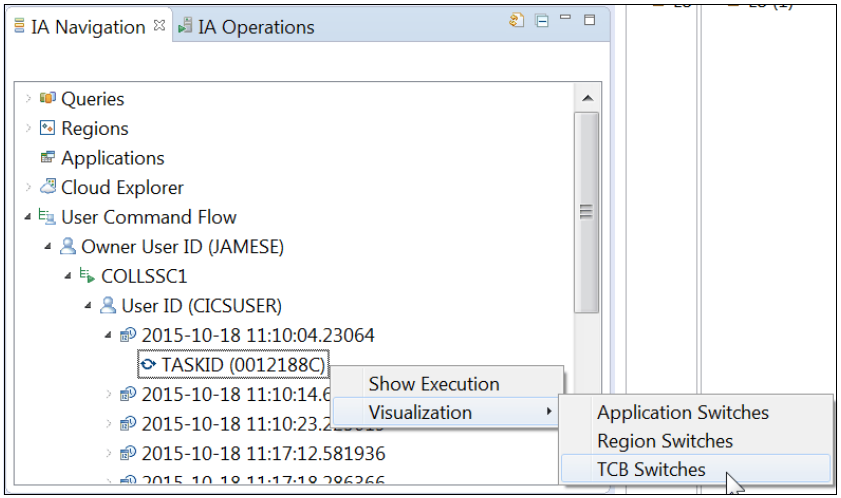


Figure 8-33 Select visualization by TCB switches

This action opens the Command Flow Diagram view as shown in Figure 8-34.

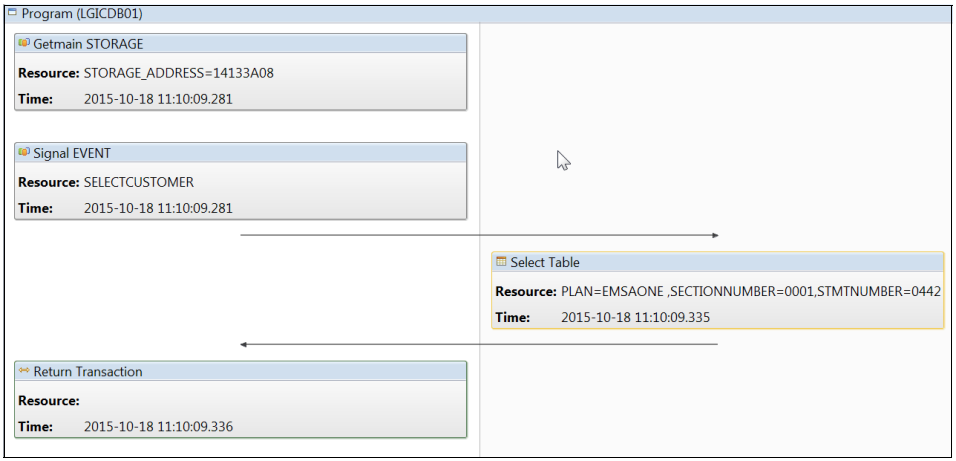


Figure 8-34 Command flow across TCB modes

In the example that is shown in Figure 8-34, you can see that the flow goes over to the L8 TCB to execute the DB2 command.

For the preceding example, we also used the **orientation** icon to change the orientation to vertical as shown in Figure 8-35.

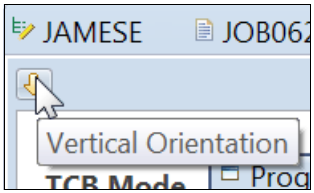


Figure 8-35 Change to vertical orientation

A

IBM Rational Asset Analyzer

So what, exactly, is Rational Asset Analyzer (IBM RAA®)? It is a DB2 database that contains just over 100 tables. It is also a set of parsers that scan source code, online resources, and web components, along with a set of programs that take the scanned information and load them into the database. During the load process, relationships are formed among the various components. Through a web browser, you can view all of the information that was scanned and collated.

To use RAA, you must first identify the production resources at your site. You then let RAA scan the resources that you want to know more about. You can scan z/OS resources and non-z/OS (that is, web-based) resources.

z/OS resources consist of source code, JCL, IMS, and CICS region information. These resources can exist in partitioned data sets or in source code change management systems (for example, SCLM or ChangeMan). Scanners for z/OS resources execute on z/OS.

Non-z/OS resources consist of Java Platform, Enterprise Edition applications (including web archive (WAR) and enterprise archive (EAR) files), Java source and bytecode, XML, HTML, and more. These resources can reside on the appropriate native file system or in
IBM Rational ClearCase®.

The distributed scanners (crawlers) for non-z/OS resources run on either Microsoft Windows 2000 or Windows NT.

After RAA stores the information about these resources in the database (note that the actual source code is not stored), the information can be shared across your enterprise by all of your application development teams.

As an IBM WebSphere® application, RAA uses JavaServer Pages (JSPs), servlets, and HTML to display information in a web browser. This interface keeps the details of the database queries hidden from view, which allows you to concentrate on the information that you seek and freeing you from the task of figuring out how to get it.

When you view the information in the database, the pages that are displayed in a web browser reflect the logical organization of the various application portfolios at your site. Through a series of links, built on the relationships among the components RAA discovered during the scan, you can drill down from the highest level of your application to a single data element. In the process, you are given visual representations of how your programs, data files, batch jobs, and transactions are related.

Rational Asset Analyzer helps your application development organization to carry out the following functions:

- Understand components and their relationships.
- Analyze the impact of a proposed change.
- Scope and develop project plans.
- Gather connector information for z/OS programs.
- Extract business logic from existing code.

RAA can be useful to a wide variety of groups in your organization that support all of the phases of the system development lifecycle (SDLC).

Members of the following groups can query the database to obtain information that can help them do a better job:

- Project managers
- Programmer analysts
- Application developers
- Quality assurance testers

They can use RAA in any phase of the following application development process:

- Requirements
- Development
- Test
- Deployment

Important: Rational Asset Analyzer requires a number of other licensed programs to support it.

For organizations that are seeking to expand their existing applications to the web, RAA provides the ability to fully explore the interrelationships among components in an application so that application development, project leaders, or group managers can prepare project plans and make the appropriate assignment of resources.

Application programmers can then use the information that was gathered initially by their team leaders to manage their workload. They can complete their assignments more quickly because of the easy way in which RAA enables them to drill down to understand the details of their application programs.

More information about Rational Asset Analyzer can be found in “Chapter 5 - Rational Asset Analyzer” in the IBM Redbooks publication “*z/OS Traditional Application Maintenance and Support*, SG24-7868”. See information about “Help from IBM” on page 240.

Using Rational Asset Analyzer with the CICS IA Explorer

Rational Asset Analyzer now provides a IBM Rational Developer for z Systems™ plug-in. In this section, we see how you leverage both the CICS IA data and the RAA data within a Rational Developer for z Systems environment.

You can download the RAA plug-in for Rational Developer for z Systems as a fix pack at the following link:

`http://www.ibm.com/support/fixcentral/swg/selectFix?product=ibm%2FRational%2FIBM+Rational+Asset+Analyzer&fixids=RAA-6.1.0-EclipsePluginForRDz&source=dbluesearch&function=fixId&parent=ibm/Rational`

After it is installed, you need to configure some preferences.

First, configure the CICS IA connection to the RAA server on the host. Click **Windows** → **Preferences** and select the `CICS Interdependency Analyzer` folder. Enter the name of the server and the port number as shown in Figure A-1.

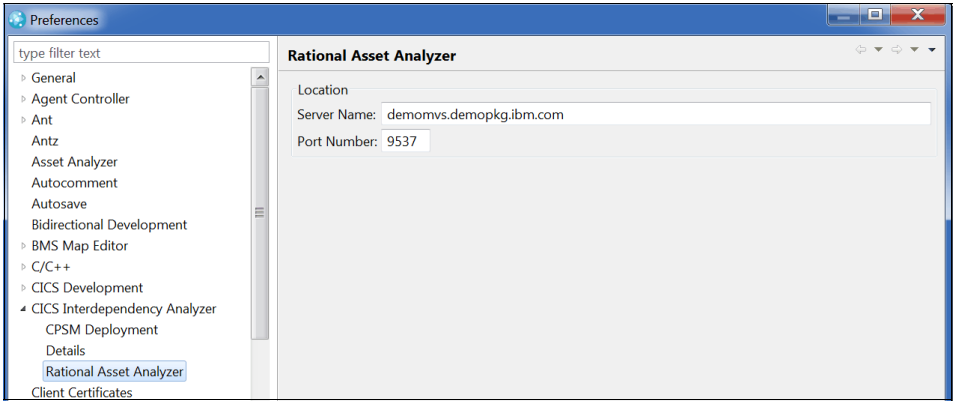


Figure A-1 CICS IA connection to the RAA server

Now configure the Rational Developer for z Systems plug-in connection to the server. Click **Windows** → **Preferences** and select the `RAA` folder. Add an entry for the server name and the TCP/IP port as shown in Figure A-2.

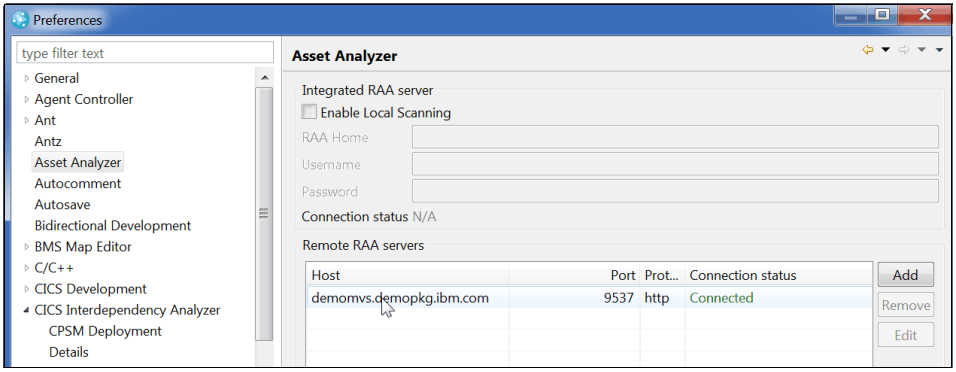


Figure A-2 Rational Developer for z Systems plug-in connection to the RAA server

For the RAA plug-in to work, you also need to connect to your host by using an RSE connection as shown in Figure A-3.

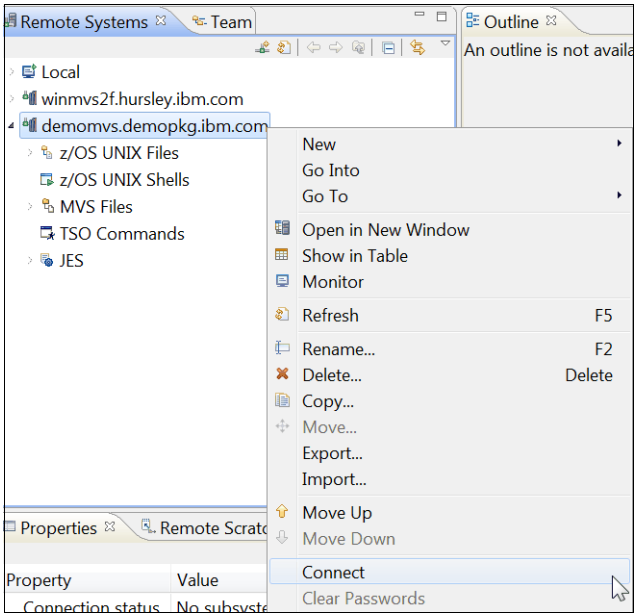


Figure A-3 Connect to remote system

IA and RAA integration before the plug-in

After you set up your CICS IA connection to RAA as shown in Figure A-1 on page 228, you can then invoke the RAA web browser from within the IA perspective.

You can right-click a transaction or program and select **Asset details** as shown in Figure A-4.

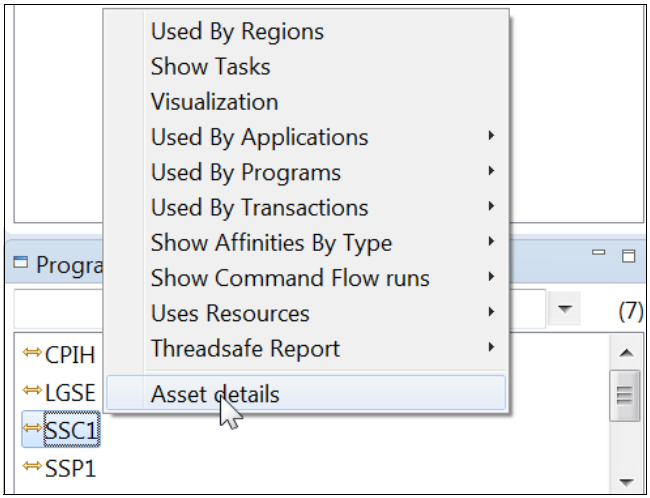


Figure A-4 Link to RAA web browser

This action opens the RAA web user interface and display information for the RAA information for transaction SSC1, as shown in Figure A-5.

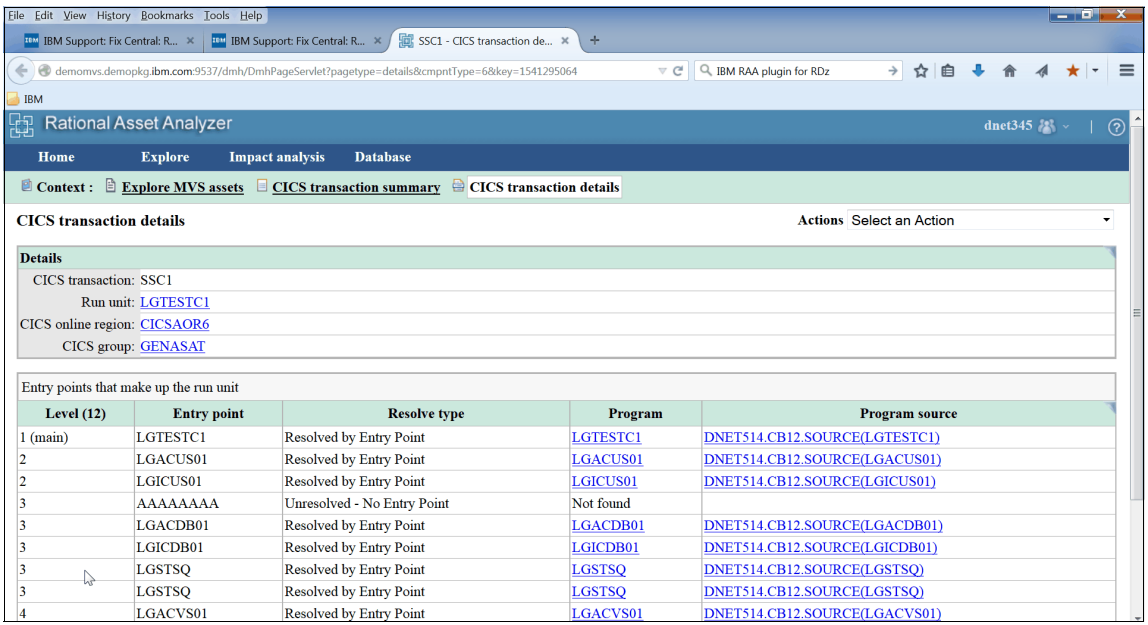


Figure A-5 RAA information for transaction SSC1

IA, Rational Developer for z Systems, and the RAA plug-in

The RAA plug-in is a search engine. It allows you to search the RAA database for specific objects such as programs and transactions.

To start a search, you can select the torch icon in the toolbar and select the “Asset Analyzer” search option in the toolbar as shown in Figure A-6.

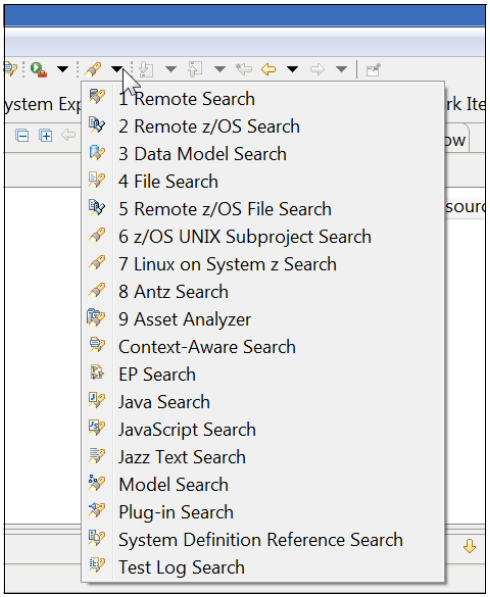


Figure A-6 Start an RAA search

Then, enter your search criteria. In this case, we search for program LGACDB01 as shown in Figure A-7.

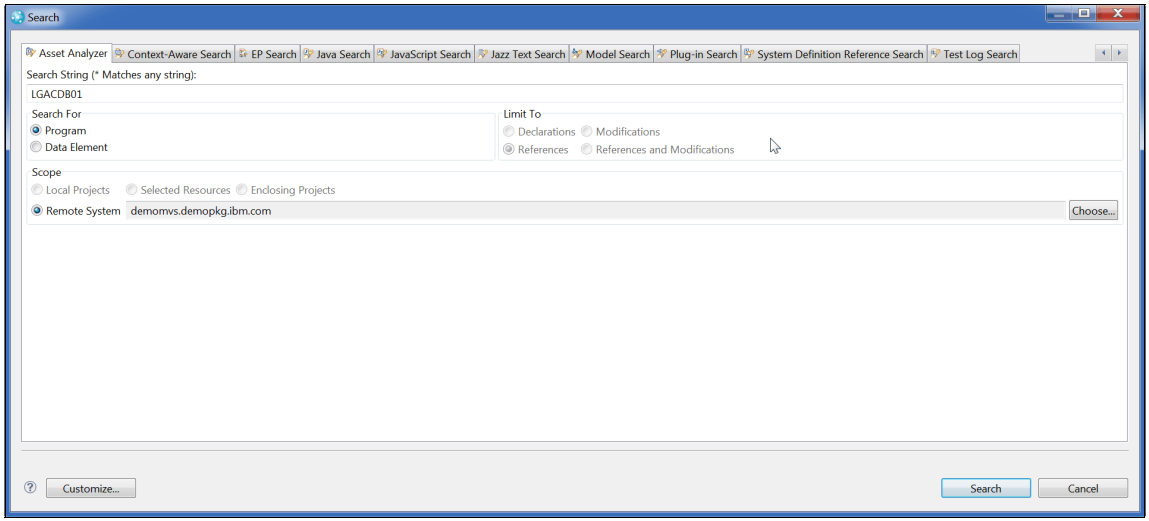


Figure A-7 Search for program LGACDB01

The search results are shown in the Search view as shown in Figure A-8.

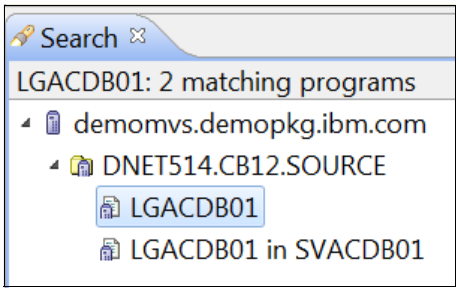


Figure A-8 Search results for program LGACDB01

From here, you have access to all the RAA plug-in features. To see what is available, right-click the program as shown in Figure A-9 on page 233.

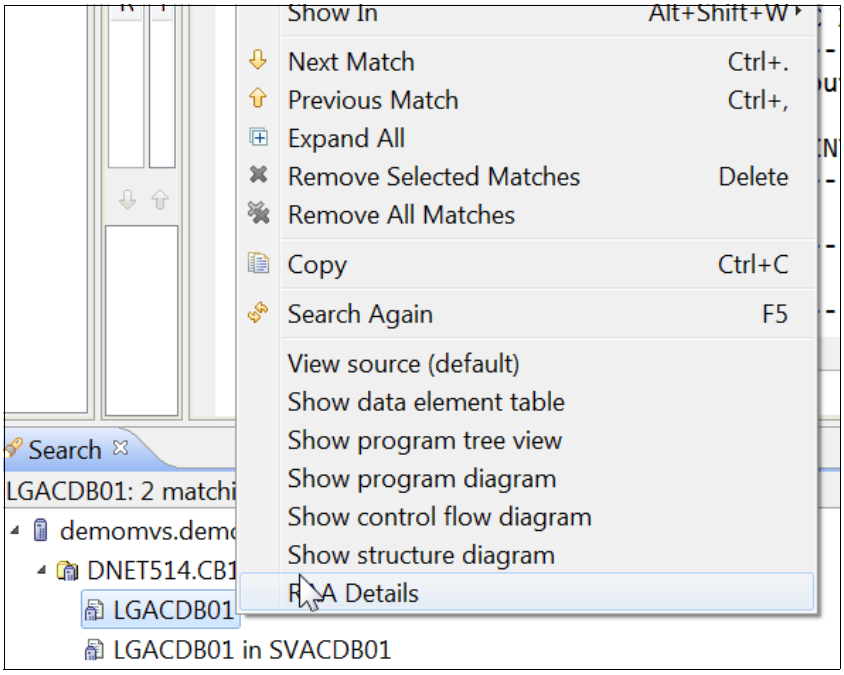


Figure A-9 RAA plug-in options for a program

To view the source, select **View source** and the source is opened in a new editor view as shown in Figure A-10.

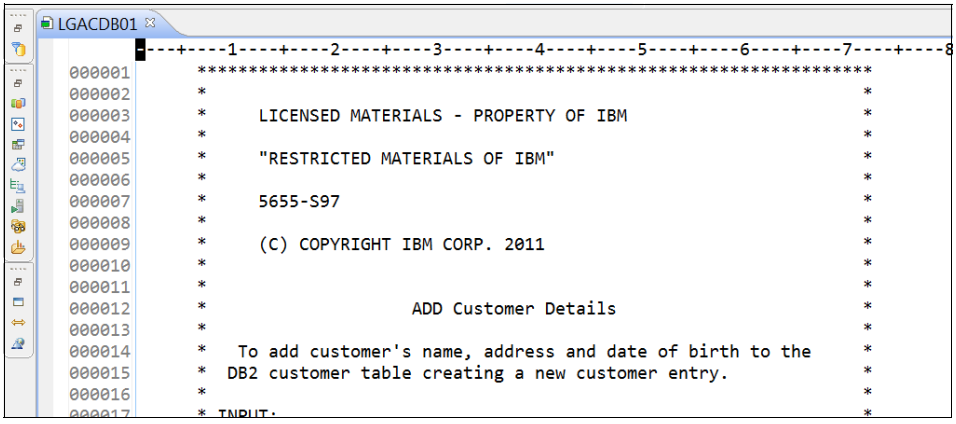


Figure A-10 Source view for program LGACDB01

B

Task collection frequency: Performance results

In CICS Interdependency Analyzer (IA) V5.2, we introduced the capability of collecting dependency data for every nth task. For more information about this feature, see 2.1.1, “Using the Optimum Collection option” on page 24.

When developing this feature, we asked our CICS performance team in Hursley to perform some measurements against the new feature. This appendix contains a brief description of the tests and the results.

Test environment

The tests were performed with an internal workload, which consisted of the following specifications:

- A COBOL BMS application with VSAM IO
- Running in two TORs routing to two AORs routing to an FOR
- With a constant transaction rate of 3800 transactions per second
- On zEC12 HA1 system, which is equivalent to a 2827-716

The following software was used:

- z/OS V2.1
- CICS TS V5.2
- CICS IA V.2

The following CICS IA options were used:

- The CICS IA collection files were shared by using VSAM RLS
- The collector was set up to collect interdependency data
- The “usage count” option was switched on
- Dynamic calls were monitored
- All CICS APIs and SPIs were monitored

The following test measurements were used:

- Used IBM RMF™ data to measure the overall CPU and transaction rate at 5-minute intervals
- The CICS IA collection frequency was varied by using the “Trigger for Task collection” option

Test results

The chart in Figure B-1 on page 237 shows the processor usage of running the CICS IA collector. The column on the left shows the CPU with CICS IA switched off. The column on the right shows the CPU processor usage for when collecting dependency data for every task. The columns in the middle show the processor usage when collecting data for every “n” task where n is 5, 10, 50, 500, and 9999. You can see that when we set the “Trigger for Task collection” value to 50 or greater, there is no great difference in the processor usage.

Note: These test results are based on an application workload that consists of EXEC CICS API and SPI calls with a minimum of business logic. This processor usage is not typical of a customer environment where an API call would be followed by business logic.

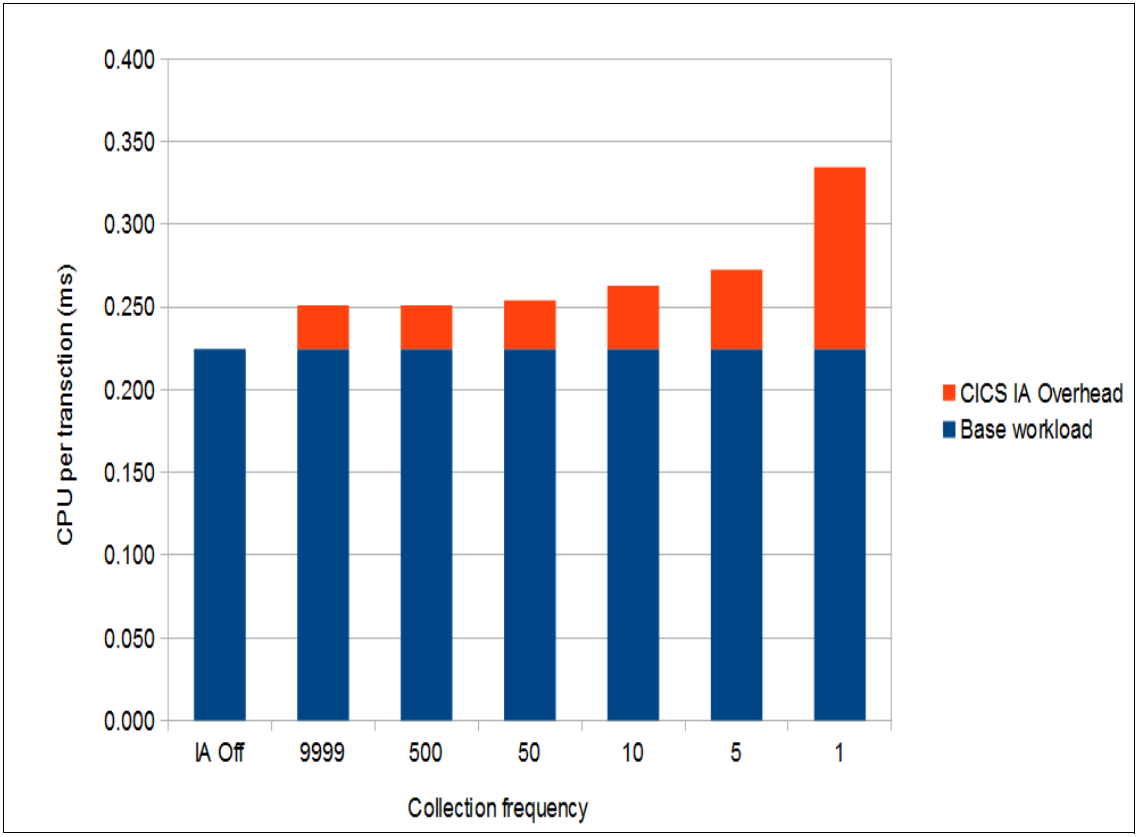


Figure B-1 CICS IA processor usage for the sample workload

The results are also shown in Table B-1.

Conclusions

The following conclusions were reached:

- Enabling CICS IA dependency collector always has an associated cost of driving the user exit, regardless of the collection frequency.
- For this sample workload, reducing the task collection frequency to lower than 1 in 50 has little or no effect.
- Collecting data for one in every 10 tasks can reduce the CICS IA processor usage by 65%.

The processor usage for the sample workload is shown in Table B-1.

Table B-1 CICS IA processor usage for the sample workload

Collect every 'n'th task	Base workload (CPU ms)	CICS IA processor usage (CPU ms)	Savings in CPU
IA Off	0.225	0.000	-
9999	0.225	0.027	76%
500	0.225	0.026	76%
50	0.225	0.030	73%
10	0.225	0.038	65%
5	0.225	0.048	56%
1	0.225	0.110	-

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *Cloud Enabling IBM CICS, SG24-8114*
- *Event Processing with CICS*, SG24-7792
- *Threadsafe Considerations for CICS, SG24-6351*
- *z/OS Traditional Application Maintenance and Support, SG24-7868*

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- *CICS IA User's Guide and Reference*, SC34-6365
- *CICS Transaction Server for z/OS Application Programming Reference*, SC34-6819
- *CICS Transaction Server for z/OS System Programming Reference*, SC34-6820

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

IBM CICS Interdependency Analyzer

SG24-6458-02

ISBN 0738441171

IBM

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

SG24-6458-02

ISBN 0738441171

Printed in U.S.A.