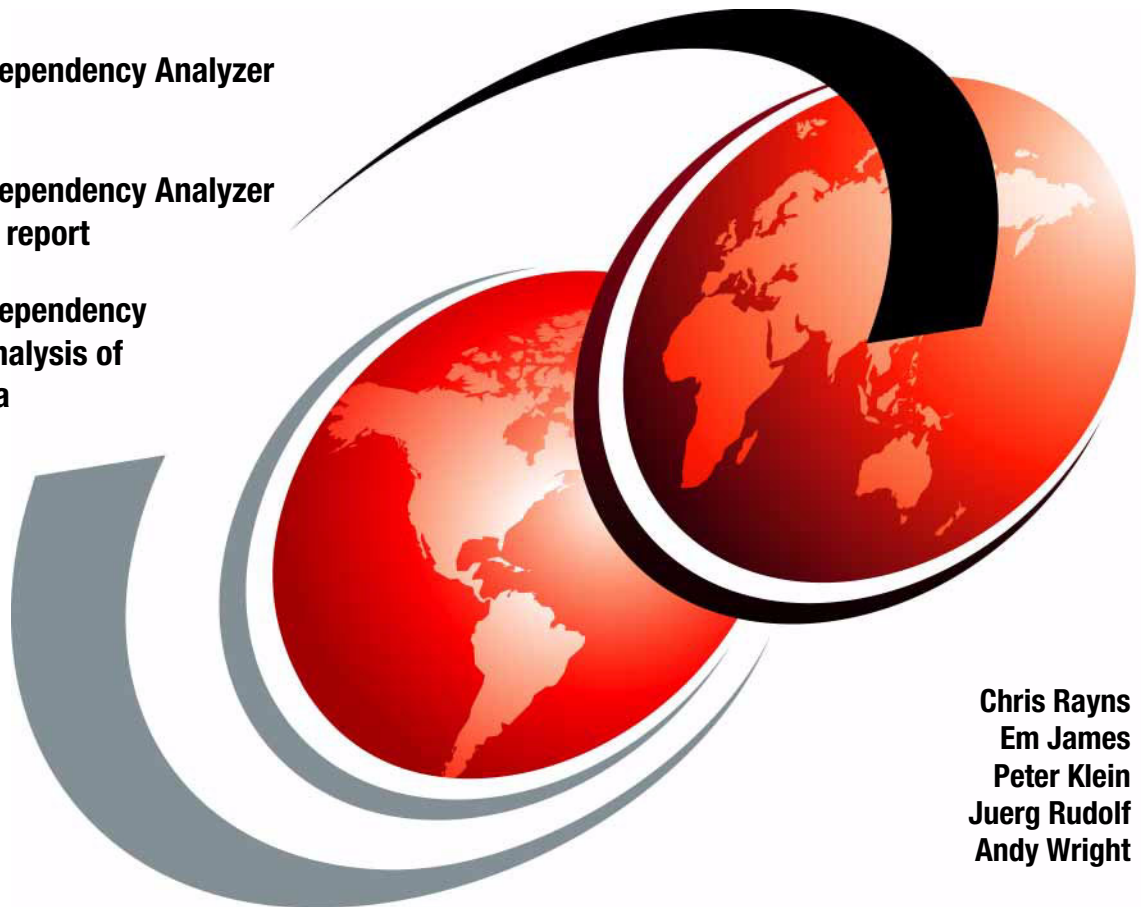


IBM CICS Interdependency Analyzer

CICS Interdependency Analyzer
Explorer

CICS Interdependency Analyzer
Threadsafe report

CICS Interdependency
Analyzer Analysis of
affinity data



Chris Rayns
Em James
Peter Klein
Juerg Rudolf
Andy Wright



International Technical Support Organization

CICS Interdependency Analyzer

September 2008

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

Second Edition (September 2008)

This edition applies to Version 2, Release 2, CICS Interdependency Analyzer.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	ix
Trademarks	x
Preface	xi
The team that wrote this book	xi
Become a published author	xii
Comments welcome	xiii
Summary of changes	xv
September 2008, Second Edition	xv
Chapter 1. CICS IA overview	1
1.1 What can CICS IA do for you	2
1.1.1 CICS IA highlights	2
1.1.2 What questions does CICS IA answer	3
1.2 What is CICS IA	3
1.3 CICS IA architecture and components	6
1.3.1 CICS IA components	6
1.4 Product information	9
Chapter 2. Installation and customization	11
2.1 CICS IA requirements	12
2.2 Using the CICS IA customization function	12
2.2.1 Running the installation and customization program	13
2.2.2 Creating the VSAM files	15
2.2.3 Defining resources to CICS	17
2.2.4 Tailoring the CICS startup job	18
2.2.5 Restarting your CICS region	19
2.3 Customizing the DB2 environment	19
2.3.1 Defining the database	20
2.3.2 Editing job CIUBND and SCIUSQL member CIUGRANT	23
2.3.3 Editing SCIUSQL member CIUGRANT	25
2.3.4 Creating the default and IVP application	26
2.4 Running the installation verification programs	30
2.4.1 Running the CIUV transaction	30
2.4.2 Running the CIUIVPLD job	30
2.5 Updating the Dependency database	33
2.5.1 Installing the CICS IA Explorer	46

Chapter 3. Scanner component	53
3.1 Creating a summary report	54
3.1.1 Contents of a summary report.	55
3.1.2 Creating a summary report with DB2 output	57
3.2 Creating a detailed report	61
3.2.1 Contents of a detailed report.	62
3.2.2 Creating a detailed report with DB2 output	63
3.3 Running the CSECT Scanner	67
Chapter 4. The Collector	73
4.1 What does CICS IA monitor	74
4.1.1 Collector components	77
4.2 The CINT transaction	77
4.2.1 The CINT menu.	78
4.3 What happens after the Collector is started	88
4.4 The DB2 load programs	89
Chapter 5. Query interface	91
5.1 The main window after CINQ is entered	92
5.2 What transactions is a given program invoked by	92
5.3 What transactions access a particular file and how	95
5.4 What are the resources that a specific program uses.	98
5.5 How is a file accessed by a particular program.	103
5.6 Which affinities does a transaction have.	105
5.7 How would you query on DB2, WMQ, or IMS	112
Chapter 6. Analyzing CICS IA data using the CICS IA Explorer	121
6.1 CICS IA Explorer overview	122
6.2 Toolbar searches and queries.	123
6.2.1 Toolbar searches	123
6.2.2 Queries	124
6.3 Programs and Transactions windows	125
6.3.1 Searching programs and transactions	126
6.4 Regions window	127
6.5 Supplied queries with CICS IA Explorer	128
6.6 Creating your own queries in CICS IA Explorer	134
6.6.1 Adding filters to the query	137
6.6.2 Modifying an existing query.	139
6.6.3 Saving your queries in folders.	141
6.6.4 Creating a query folder	141

6.7 Analyzing query and search results	142
6.8 Comparing query and search results	144
6.9 Using WebSphere Studio Asset Analyzer with CICS IA Explorer	146
Chapter 7. Sample reports and queries	151
7.1 Affinities	152
7.1.1 V_CIU_AFFINITY	152
7.2 Dependencies	156
7.2.1 CIU_CICS_DATA	156
7.2.2 V_CIU_CICS_INDS	165
7.2.3 CIU_DB2_DATA	167
7.2.4 CIU_IMS_DATA	172
7.2.5 CIU_MQ_DATA	173
7.3 Resources	175
7.3.1 CIU_EXIT_INFO	175
7.3.2 CIU_FILE_DETAIL	176
7.4 Scanner base tables	180
7.4.1 CIU_SCAN_SUMMARY	180
Chapter 8. CICS IA usage scenarios	183
8.1 Threadafety analysis	184
8.1.1 Background to the CICS Open Transaction Environment	184
8.1.2 Using CICS IA to assist with threadafety analysis	187
8.1.3 Using the CICS IA Explorer to assist with threadafety analysis ..	193
8.2 CICS Web services assistance	195
8.2.1 Web services requester detection	196
8.2.2 Web services provider detection	197
8.2.3 Identifying candidate Web services programs	197
8.3 Migration assistance	199
8.3.1 Migration drivers and inhibitors	199
8.3.2 CICS IA application migration support	200
8.3.3 Identifying TRUEs and GLUEs	201
Chapter 9. Affinities	203
9.1 Overview of transaction affinities	204
9.1.1 Inter-transaction affinity	204
9.1.2 Transaction-system affinity	205
9.1.3 Affinity relations	205
9.1.4 Affinity lifetimes	206
9.2 The Affinities Scanner and the Affinities Collector	207
9.2.1 The Affinities Scanner	207
9.2.2 The Affinities Collector	208
9.2.3 The Affinities Reporter	210
9.2.4 The Builder	210

9.2.5 Detected API and SPI commands	210
9.3 Customizing and running the Affinities Scanner	212
9.3.1 The summary report	212
9.4 Running the Affinities Collector	214
9.5 Customizing and running the Affinities Reporter	215
9.5.1 Tips for analyzing reports	220
9.5.2 How to proceed	222
9.6 Customizing and running the Builder	223
9.7 Uploading the data into CICSplex SM	226
Chapter 10. Hints and tips	231
10.1 Collector performance	232
10.1.1 What resources to collect	232
10.1.2 Excluding programs or transactions from the collector	239
10.1.3 Using the Timer options	242
10.1.4 Other General Options that can affect performance	244
10.2 Transaction CINB performance	247
10.2.1 Database load performance	248
10.2.2 Database query performance	249
10.3 IA database cleanup	250
Chapter 11. Debugging	257
11.1 Installation and customization	258
11.1.1 What to look for if something goes wrong	258
11.2 The Collector component	258
11.2.1 What exits are invoked	258
11.2.2 What to look for if Collection is not happening	260
11.3 CINC transaction	260
11.3.1 Resolving DB2 SQL error codes	261
11.4 The CICS IA Explorer	262
11.4.1 What level of the CICS IA Explorer am I using	262
11.4.2 What if I get SQL error code -805	262
11.5 CICS IA Trace Facility	263
11.6 Taking a dump of CICS IA	264
11.7 The plan table	265
Appendix A. WebSphere Studio Asset Analyzer	267
WSAA core functions	269
Inventory collection	270
Application exploration	271
Connector information	272
Help	272
Terminology	272
Site	273

Application	273
Concatenation set	274
Asset	274
Using WebSphere Studio Asset Analyzer with CICS IA Explorer	275
Related publications	277
IBM Redbooks	277
Other publications	277
How to get Redbooks	277
Help from IBM	278
Index	279

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

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

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

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

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](http://www.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:

C/370™	IMS™	Redbooks (logo)  ®
CICSplex®	Language Environment®	System z®
CICS®	MVS™	VTAM®
ClearCase®	OS/390®	WebSphere®
DB2 Universal Database™	RACF®	z/Architecture®
DB2®	Rational®	z/OS®
IBM®	Redbooks®	

The following terms are trademarks of other companies:

J2EE, Java, JavaServer, JDBC, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

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

Preface

The IBM CICS® Interdependency Analyzer (CICS IA) is a runtime tool for use with CICS Transaction Server for z/OS®. It has two main purposes:

- ▶ To identify the sets of resources that are used by individual CICS transactions and their relationships to other resources.
- ▶ To collect and analyze data about transaction affinities. Transaction affinities require particular groups of transactions to be run either in the same CICS region or in a particular region.

In this IBM® Redbooks® publication, we first provide a detail overview of what CICS IA is and what business issues it addresses before we review the installation and customization of the product.

We discuss the scanner and collector components in detail, and we then focus on the query interface and the new CICS IA Explorer.

The team that wrote this book

This book was produced by a team of specialists from around the world working at Mainz Germany.

Chris Rayns is an IT Specialist and the CICS project leader at the International Technical Support Organization, Poughkeepsie Center. He writes extensively on all areas of CICS. Before joining the ITSO, Chris worked in IBM Global Services in the United Kingdom as a CICS IT Specialist.

Em James is a Technical Specialist for CICS Tools working at IBM Hursley Labs in the United Kingdom. He has 18 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.

Peter Klein is a CICS Team Leader at the IBM Germany Customer Support Center. He has 18 years of experience working as a Technical Support Specialist with IBM software products. His expertise includes WebSphere® MQ, CICSPlx® System Manager, and distributed transaction systems. Peter has contributed to several other IBM Redbooks publications and ITSO projects that IBM Learning Services sponsored.

Juerg Rudolf is an IT-Architect working in the Application Integration Team of IBM Global Business Services in Zurich, Switzerland. He has 35 years of experience in IT. His areas of expertise include CICS, CICS performance, and application integration and design.

Andy Wright is a Senior Software Engineer working in the CICS Change Team at the IBM Hursley Laboratory in the United Kingdom. He has a Bachelors degree in Physics and Computing from Southampton University and a Masters degree in Software Engineering from the University of Oxford. He has 20 years of experience with CICS and related CICS products. He is the author of over 70 technical articles and papers on CICS software and debugging and diagnostic techniques. Andy also presents on CICS topics at conferences in the United States and Europe.

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

Darren Beard
IBM Hursley

John Knutson
IBM Hursley

Richard Conway
International Technical Support Organization, Poughkeepsie Center

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

Authors of the first edition, CICS IA, published in 16 November 2004, were:

- ▶ Thomas Braun
- ▶ James Matson
- ▶ Dave White

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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 in one of the following ways:

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

ibm.com/redbooks

- ▶ Send your comments in an e-mail 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

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 SG24-6458-01 and CICS IA, as created or updated on September 15, 2008.

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



CICS IA overview

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

- ▶ What business issues does CICS IA address?
 - CICS IA highlights
 - What questions does CICS IA answer?
- ▶ What is CICS IA?
- ▶ CICS IA architecture and components

1.1 What can CICS IA 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.

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 may 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.

Maintenance or enhancement of applications

During the normal application life cycle, 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.

1.1.1 CICS IA highlights

Here are some of the benefits of using CICS IA:

- ▶ Automate detection of runtime resource relationships
- ▶ Understand application flow with flexible resource-relationship reports
- ▶ Accumulate resource-relationship data in a DB2® database
- ▶ Build relationship maps to help improve the speed of application maintenance and to help reduce the time to resolve problems
- ▶ Compare applications and resources across regions
- ▶ Migrate, reuse, and extend CICS applications more quickly

- ▶ Support SOA implementations with deep application understanding
- ▶ Make informed decisions about the best way to split workloads
- ▶ Minimize the impact of routine application maintenance for the user

1.1.2 What questions does CICS IA answer

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(s) does a particular CICS application run in?
- ▶ What are all of the CICS resources that a given application can use?
- ▶ What are all of the CICS resources that a given transaction can use?
- ▶ What transactions belong to a given application?
- ▶ What programs does a given transaction invoke?
- ▶ What transactions access a particular file and how?
- ▶ What resources does a specific program use?
- ▶ How is a file accessed by a particular program?
- ▶ Which affinities does a transaction have?
- ▶ How do you query on DB2, MQ, or IMS™ resource use?

1.2 What is CICS IA

IBM CICS Interdependency Analyzer for z/OS is the IBM discovery tool for CICS Transaction Server for z/OS, which offers a wide range of new capabilities and supports completely new functionality in CICS Transaction Server, Version 3.2. New capabilities, such as threadsafe and affinity analysis, a new user interface, and support for Software AG Natural, make it possible to achieve better reuse, management, and control of your applications through improved understanding of an even wider range of CICS systems and resources. CICS Interdependency Analyzer facilitates projects, such as CICS version-to-version migration, affinity removal, and Web service refactoring that depend, for their success, on deep knowledge of application, system, and resource relationships.

CICS IA is a runtime and batch system to use with CICS Transaction Server for z/OS and CICS Transaction Server for OS/390®. It has two purposes:

- ▶ To identify CICS application resources and their interdependencies

This function 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)
- Which resources are no longer used

- What applications does a CICS region contain

CICS IA captures interdependency information while CICS is running and stores this information in VSAM files. You can produce detailed reports from the VSAM files, if you want to. Subsequently, the VSAM files are used to load the DB2 database.

- ▶ To analyze transaction affinities

Affinities require particular groups of transactions to be run either in the *same* CICS region or in a *particular* region.

Affinities 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.

CICS IA captures and loads the Affinity data into its DB2 databases. You can then query the data with the CICS IA Explorer, the CICS IA online transaction CINQ, with batch SQL queries, or with other SQL query software tools. Using batch SQL processing, you can produce detailed reports.

Many large organizations have used CICS since the early 1970s, and their systems grow and evolve with the business. During this time, many techniques of implementing applications were used — as a result of new function, changing corporate standards, technical requirements, and business pressures.

Frequently, this growth was not as structured as it could have been, and the result was that many applications and services share common resources and changes in one area that typically affect many others, which can reach such a level that the system can no longer develop in a controlled manner without a full understanding of these interrelationships. CICS IA can help you achieve this understanding, for example, if you need to change the content or structure of a file, you need to know which programs use the file because you must change those programs too. CICS IA can tell you which programs are affected and can also give you the transactions that drive the programs. CICS IA records the interdependencies between resources (such as files, programs, and transactions) by monitoring programming commands that operate on resources.

The application that issues such a command has a dependency on the resource that is named in the command, for example, if an application program issues the command EXEC CICS WRITE FILE (*myfile*), it has a dependency on the file called *myfile*. It might have similar dependencies on transient data queues, temporary storage queues, transactions, other programs, and so on.

The commands that are monitored are typically CICS application programming interface (API) and system programming interface (SPI) commands that operate

on CICS resources. However, you can also instruct CICS IA to monitor some types of non-CICS command that operate on non-CICS resources, for example:

- ▶ MQ calls to WebSphere MQ resources
- ▶ EXEC DLI calls to IMS database resources
- ▶ DB2 calls
- ▶ Dynamic COBOL calls to other programs

Potentially, the inclusion of any non-CICS resources gives you a fuller picture of the resources that a transaction uses.

The Collector component of CICS IA collects the dependencies that apply to a single CICS region; that is, an AOR or a single, combined routing region-AOR. You can run the Collector component of CICS IA against production CICS regions and also use it in a test environment to monitor possible dependencies that new or changed application suites or packages introduce.

From the interactive interface of CICS IA, you can control Collectors that run on multiple regions.

Note: To ensure that you monitor as many potential dependencies as possible, use CICS IA with all parts of your workload, including rarely-used transactions and abnormal situations. CICS IA collects these dependencies into a database. You can store the dependency information from several CICS regions into the same database. You can review the collected dependencies using the CICS IA Query interface or list them using the Reporter.

Enhancements in CICS IA Version 2 Release 2

A range of significant enhancements are now delivered in CICS IA V2.2, which includes:

- ▶ 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

- ▶ Enhanced information about IBM WebSphere MQ calls
- ▶ Information that identifies which CICS Global User exits, and Task Related User exits are exploited in any CICS region

1.3 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.

1.3.1 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 will use. An analysis of these calls provides a view of resource interdependencies. CICS IA also captures resource affinity information.

Figure 1-1 on page 7 shows the components of CICS IA. In upcoming chapters, we cover these components in detail.

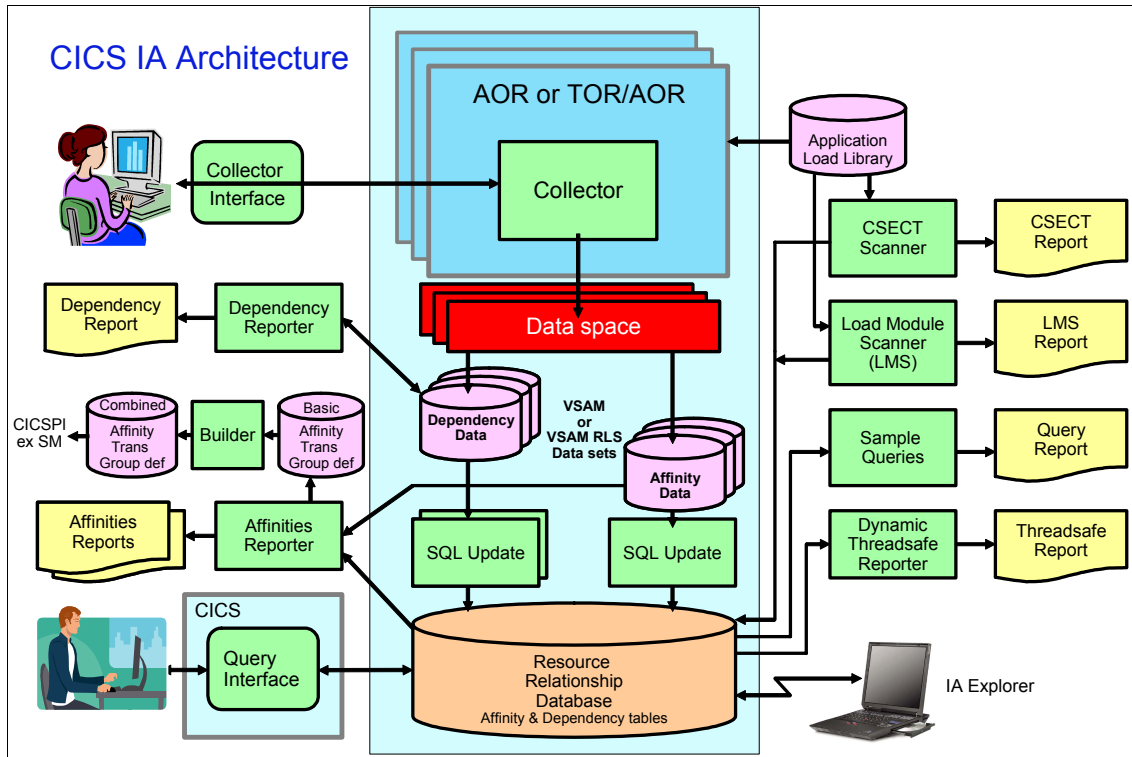


Figure 1-1 CICS IA component architecture

The Load Module Scanner

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.

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 in conjunction with the DB2 dependency tables, to identify different versions of programs. For information about using the CSECT Scanner, see Chapter 3, “Scanner component” on page 53.

The Collector

You can use the *Collector* in real-time to detect transaction resource definitions and transaction affinities in a running CICS region.

For performance reasons, you cannot collect both dependency data and affinity data on the same region, at the same time. You can collect, simultaneously, dependency data on one region and affinity data on another.

The Collector saves details of the dependencies or affinities in an z/OS data space. This data is subsequently saved to VSAM files. The Collector consists of:

- ▶ A control transaction, CINT
- ▶ An autosave transaction, CINB
- ▶ Some global user exit programs
- ▶ A task-related user exit program

We discuss the Collector in detail in Chapter 4, “The Collector” on page 73.

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, refer to the *CICS IA Users Guide*, which is in the CICS Infocenter:

<https://winlnx0c.hursley.ibm.com/19000/help/index.jsp>

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 Chapter 8, “CICS IA usage scenarios” on page 183.

The CINC Query

Subsequently, the VSAM files are loaded into a DB2 database. When the data is available in the DB2 database, you can use the *Query* component to view resource interdependencies. This component is comprised of a set of CICS transactions (COBOL/BMS).

The Explorer

The CICS IA *Explorer* is a standalone PC application that includes an Eclipse Rich Client platform and Java™ Runtime Environment. It Connects to the CICS IA DB2 relationship repository on System z® using JDBC™ Type 4 drivers. Access requires a user ID and password on the host.

1.4 Product information

CICS IA is a one-time-charge (OTC) product and is *not* included as a part of CICS TS. The program product number for CICS IA is 5697-J23.

Hardware Requirements:

- ▶ CICS Interdependency Analyzer Version 2.2 runs on any IBM z/Architecture® server on which the applicable operating system and software will run.
- ▶ CICS Interdependency Analyzer Explorer runs on any workstation that supports Microsoft® Windows® XP, Microsoft Windows 2000, or Microsoft Windows Vista® with 256 MB RAM minimum and 80 MB of free disk space. For mainframe connectivity for DB2 data analysis, the workstation must be configured for TCP/IP network access.

Software Requirements:

- ▶ CICS Interdependency Analyzer Version 2.2 is designed to be used with CICS Transaction Server for z/OS Version 3.2 (5697-MI5). It can also be used with CICS Transaction Server for z/OS Version 3.1 or CICS Transaction Server for z/OS Version 2.2 or Version 2.3 (5697-E93), but at a reduced level of function.
- ▶ CICS Interdependency Analyzer Version 2.2 runs with any supported level of operating system with which the applicable CICS Transaction Server runs.
- ▶ CICS Interdependency Analyzer Version 2.2 requires IBM z/OS SMP/E for installation and maintenance.
- ▶ CICS Interdependency Analyzer Version 2.2 requires access to an IBM DB2 Universal Database™ server for IBM OS/390 and z/OS Version 7.1 (5675-DB2) or later.

- ▶ CICS Interdependency Analyzer Explorer requires Microsoft Windows XP, Microsoft Windows 2000, or Microsoft Windows Vista.
- ▶ DB2 Utilities Suite for z/OS Version 8 (5655-K61) or later.



Installation and customization

In this chapter, we provide steps for you to take before you can use CICS IA.

The SMP/E installation process for CICS IA is described in the Program Directory that is distributed with the product.

In this chapter, we describe the simplest way to install and customize the CICS Interdependency analyzer in a single CICS region.

We do not cover migration steps from an older release. For this, refer to the *CICS Interdependency Analyzer for z/OS User's Guide and Reference Version 2 Release 2*, SC34-6790.

2.1 CICS IA requirements

To use CICS IA you need:

- ▶ z/OS Version 1.07.0 or later (program number 5694-A01).
- ▶ CICS Transaction Server:
 - CICS Transaction Server for z/OS 2.2 or later
 - CICS Transaction Server for z/OS 3.1 or later
- ▶ Each CICS region on which the CICS IA Collector is to run must have Language Environment® installed and active. A CICS region on which the CICS IA Query interface is to be run must have DB2 installed and active. Any of the following DB2 product numbers are required or CICS IA will not install on your system:
 - 5675-DB2 DB2 Universal Database Server for OS/390 V7.1 or later
 - 5625-DB2 DB2 Universal Database Server for OS/390 V8.1 or later
 - 5635-DB2 DB2 Universal Database Server for OS/390 V9.1 or later
- ▶ To control CICS IA Collectors on multiple regions from a single CICS terminal, the VSAM files to which CICS saves dependency data and control information must be shared across all of the regions. To share these files, you can use either of the following items:
 - VSAM record-level sharing (RLS): If you use VSAM RLS, all regions must be in the same z/OS parallel sysplex. (A parallel sysplex is a sysplex that uses a coupling facility, which is required to support VSAM RLS).
 - Function shipping to a file-owning region (FOR): For information about CICS function shipping, see *CICS Intercommunication Guide*.
- ▶ If you are using DB2 Version 8 or later, you require: DB2 Utilities Suite for z/OS Version 8 (5655-K61) or later.

2.2 Using the CICS IA customization function

In this section, we describe the steps that you must take before you can use CICS IA.

We recommend that you use the CICS IA installation and customization program to create a complete set of customized installation jobs. The installation and customization program does not change existing original libraries. It creates a new set of libraries that contain customized installation jobs, SQL definitions, and lists. Figure 2-1 on page 13 shows the basic concept of how the installation and customization program assists you to customize your CICS IA sample jobs and sample SQL.

2.2.1 Running the installation and customization program

You can run member CIUCNFG1 of the SCIUEXEC library to invoke the installation and customization program. In order to execute the program, we enter the command in Example 2-1 at the ISPF command shell.

Example 2-1 CIUCNFG1 invocation

```
exec 'CIU.V2R2M0.SCIUEXEC(CIUCNFG1)' 'CIU.V2R2M0 ENU'
```

The installation and customization program requires two parameters to be passed to it:

- ▶ The high-level qualifier of the CICS IA data sets. We use CIU.V2R2M0.
- ▶ The national language to be used. We use ENU.

Figure 2-1 illustrates the CICS IA customization function.

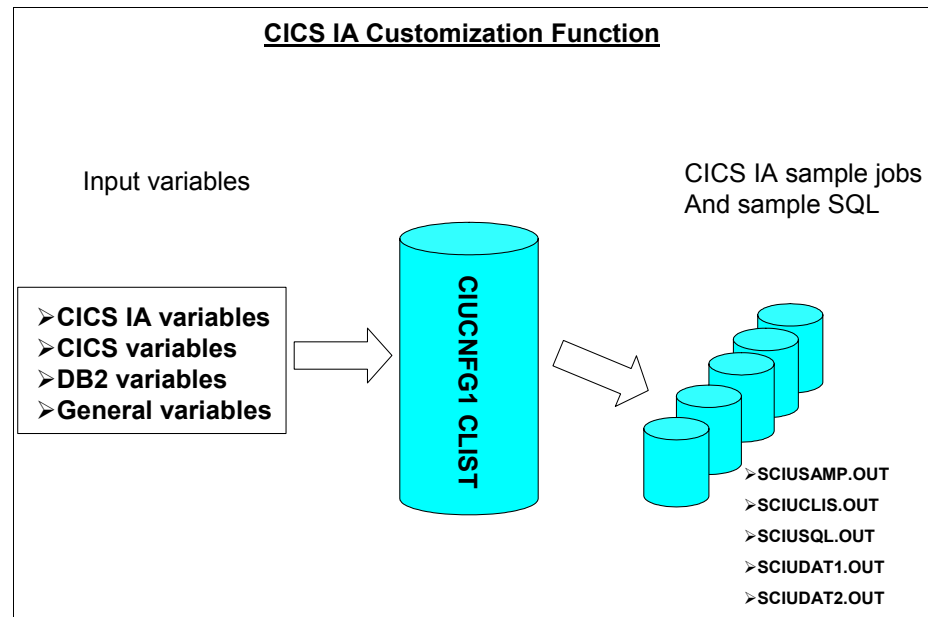


Figure 2-1 CICS IA customization function

Program CIUCNFG1 takes a range of the variables in the following list to create customized sample jobs. There are five sections where you can specify variables:

- ▶ CICS IA variables: In this section, we specify the high-level qualifier of the CICS IA libraries and the qualifier and size of the CICS IA VSAM files. Example 2-2 shows the information that we specified:
 - We specify the CICS IA qualifier.
 - We use qualifier CICSSYSF.CICSTS32 for the CICS IA VSAM files and allocate 10 cylinders space for each file.
 - The TCP/IP PORT number is obsolete for CICS IA V2.2

Example 2-2 CICS IA variables

CICS IA variables:

```
IA PRODUCT QUALIFIER      : CIU.V2R2M0
IA VSAM FILE QUALIFIER    : CICSSYSF.CICSTS32_____
IA VSAM FILE DATACLASS   : _____
IA VSAM FILE STORAGECLS  : _____
IA VSAM FILE MNGMNTCLASS : _____
IA VSAM FILE SPACE UNITS : CYLINDERS (CYLINDERS, TRACKS, RECORDS, KILOBYTES
                                or MEGABYTES)
IA VSAM FILE PRIMARY QTY : 10_____ (In above units)
IA VSAM FILE SECNDRY QTY : 2_____ (In above units)
IA TCP/IP PORT NUMBER     : 37890
```

- ▶ DB2 variables: In this section, we provide information about the DB2 subsystem that we use to install CICS IA. Example 2-3 shows the DB2 information that we specified on our system. You might need to contact your DB2 system administrator to get the information about your DB2 subsystem.

Example 2-3 DB2 variables

DB2 variables:

```
DB2 QUALIFIER              : DB9E9
DB2 RUNLIB.LOAD HLQ        : DB9EU
DB2 PROCLIB DATASET NAME  : DB9EU.PROCLIB
DB2 SUB SYSTEM             : D9E1
DB2 TABLE QUALIFIER      : CICSRS3
DB2 TABLE OWNER          : CICSRS3
DB2 DASD VOLUME           : XXXXXX
DB2 VERSION                : V910
DB2 RACF GROUP            :
DB2 BUFFERPOOL FOR TBSPC  : BPO
DB2 BUFFERPOOL FOR INDEX  : BPO
DB2 STOGROUP FOR TBLSPCS  : CIUSGTS
```


DB2 STOGROUP FOR INDEXES : CIUSGIX

DB2 DATABASE FOR TBLSPCS : CIUTBLSP
DB2 PLAN NAME FOR CICS : CIUCICS
DB2 PLAN NAME FOR BATCH : CIUBTCH

- ▶ **CICS variables:** In this section, we provide the CICS transaction server information that is required to install CICS IA. Example 2-4 shows the information we use on our system.

Example 2-4 CICS variables

CICS variables:

CICS VERSION : V320
CICS QUALIFIER : CICSTS32.CICS
CICS CSD FILE NAME : CICSSYSF.CICSVS32.DFHCS
CICS CSD LIST NAME : CIULIST
RLSACCESS FOR VSAM FILES : NO
LSRPOOLID FOR VSAM FILES : 1
CICS FILE OWNING REGION :

- ▶ **Migration variables:** We did not migrate from a previous version of CICS IA; therefore, we did not specify any information in this section. You can use the migration variables section to specify information about the current and target environment.
- ▶ **General variables:** In this section, we specify which assembler program and language environment qualifier we use, as shown in Example 2-5.

Example 2-5 General variables section

General variables:

ASSEMBLER PROGRAM : ASMA90
LE QUALIFIER : CEE

2.2.2 Creating the VSAM files

To create the CICS IA VSAM files, we run jobs CIUJCLCC and CIUJCLCA. A customized version of the jobs are copied to the SCUISAMP.OUT data set.

Sizing considerations:

A customer who uses CICS IA in more than 300 CICS regions had the following experience:

“One important thing we do is a twice-monthly delete/define. This gives us a fresh copy and prevents continuous growth. Several things can cause continuous growth in the repository, such as resources that use a variable in the resource name. One such example would be a program that builds an enqueue name from a record key. Nasty little things like this can exist everywhere and would eventually cause the VSAM repository to balloon out of control. What we look at and keep historically is in the DB2 table. We use the standard reorg/purge utility to maintain the correct amount of data in the DB2 repository. This enables us to discard old or unneeded records and at the same time provides a backup of that discarded data.”

Table 2-1 on page 17 shows the required CICS IA VSAM files and associated jobs.

When the installation and customization program completes successfully, you can review jobs CIUJCLCA and CIUJCLCC. Both jobs are copied to the CIU.V2R2M0.SCIUSAMP.OUT data set.

For the time being, we do not want to run the collector on multiple CICS regions. Therefore we use the customized jobs to define non-RLS files. In order to run the collector on multiple CICS regions, you must share the dependency and affinity data files and the control record file across all of the regions. To share the files, you can use VSAM RLS or function shipping of file control requests to a file owning region.

Table 2-1 CICS IA VSAM files and associated jobs

File	Description	Job
hlq.CIUAFF1	The CICS Affinities data file for keys equal to 16.	CIUJCLCA
hlq.CIUAFF2	The CICS Affinities data file for keys equal to 32.	CIUJCLCA
hlq.CIUAFF3	The CICS Affinities data file for keys greater than 32.	CIUJCLCA
hlq.CIUCNTL	The control record file. A recoverable file used to hold control information.	CIUJCLCC
hlq.CIUINT1	The CICS dependency data file. An unrecoverable file used to record dependencies on CICS resources with names up to 32 bytes long.	CIUJCLCA
hlq.CIUINT2	The DB2 dependency data file. An unrecoverable file used to record dependencies on DB2 resources.	CIUJCLCA
hlq.CIUINT3	The MQ dependency data file. An unrecoverable file used to record dependencies on MQ resources.	CIUJCLCA
hlq.CIUINT4	The IMS dependency data file. An unrecoverable file used to record dependencies on IMS resources.	CIUJCLCA
hlq.CIUINT5	The CICS +32 dependency data file. An unrecoverable file used to record dependencies on CICS resources with names longer than 32 bytes.	CIUJCLCA
hlq.CIUINT6	The resource detail data file. An unrecoverable file used to record extra detail data for CICS resources.	CIUJCLCA

2.2.3 Defining resources to CICS

Customized jobs are provided by the installation and customization program to create CICS resource definitions for CICS IA. The following resource definitions are defined:

- ▶ CICS IA program components
- ▶ CICS IA transactions CINT, CINB, and Cinq

- ▶ VSAM files CIUCNTL, CIUINT1 through CIUINT6, and CIUAFF1 through CIUAFF3
- ▶ DB2 entry (DB2ENTRY) definitions and DB2 transaction (DB2TRAN) definitions
- ▶ CINT transient data queue for messages

The following jobs are available:

- ▶ For CICS TS for z/OS Version 1.3, review and run job CIUJ13CR
- ▶ For CICS TS for z/OS Version 2.2, review and run job CIUJ22CR
- ▶ For CICS TS for z/OS Version 2.3, review and run job CIUJ23CR
- ▶ For CICS TS for z/OS Version 3.1, review and run job CIUJ31CR
- ▶ For CICS TS for z/OS Version 3.2, review and run job CIUJ32CR

We use CICS TS for z/OS Version 3.2 to install CICS IA; therefore, we review and run job CIUJ32CR to define the CICS IA resource definitions.

2.2.4 Tailoring the CICS startup job

To enable CICS IA to run in your CICS region, you have to do the following:

1. Specify the system initialization parameter DB2CONN=YES. This parameter is necessary to run the CINQ query transaction and the CICS IA Explorer.
2. If you plan to use VSAM RLS to share the Dependency data file and control record file across multiple regions, specify the system initialization parameter RLS=YES.
3. Set the ICVR system initialization parameter to at least 10 seconds, that is, ICVR=10000 (or a larger value). If you do not do this, the Collector or one of your own transactions might end prematurely with an abend code of AICA.
4. Add the following load libraries to the DFHRPL concatenation in the startup job JCL:
 - hlq.SCIULOAD
 - hlq.SCIULODE
5. Add the following DD statement for the CINT transient data message log:


```
//CINT DD SYSOUT=*
```
6. On any region on which you intend to collect DB2 data, ensure that the user ID under which the CINB transaction runs has permission to access the SISIBM.SYSPACKSTMT and SYSIBM.SYSSTMT DB2 tables. You can give access by granting access to the CICS IA plan or by adding the user ID to the RACF® group that has access. See SQL member CIUGRANT in SCIUSQL.

Note: Check with your CICS system administrator if you do not know. The user ID under which the CINB transaction runs is dependent on how CICS IA is activated using CINT. In most cases, the "CICS default user ID" is used. However, there are cases where the PLT user ID is used, for example, if started by PLT processing, the ID of the current CINT transaction or the "Link ID" if the CINT transaction is routed to another CICS region.

If you want to start and stop CICS IA from the PLT, refer to *CICS Interdependency Analyzer for z/OS User's and Reference Version 2 Release 2*, SC34-6790.

2.2.5 Restarting your CICS region

After you make all changes, restart your CICS region using the modified CICS startup job. Make sure that the new RDO definitions are installed by using the system initialization parameter `START=COLD` or `START=INITIAL`. If this is not possible, install the new RDO group after the CICS warm start with the `CEDA INSTALL GROUP(CIUxxG22)` command, where `xx` is the release qualifier of your CICS.

2.3 Customizing the DB2 environment

In this section, we tell you how to set up the DB2 environment for CICS IA. You must perform the following steps:

1. Create the database tables using the customized member `CIUDBCR` in the `hlq.SCIUSAMP.OUT` data set.
2. Bind the DBRMs using the customized member `CIUDBND` in the `hlq.SCIUSAMP.OUT` data set.
3. Authorize user IDs to access the plans using the `CIUDBND` and `SCIUSQL` member `CIUGRANT`.
4. Define the applications using the jobs `CIUANEW`, `CIULOAD`, and `CIUDESC`.

2.3.1 Defining the database

The installation and customization program copies a customized version of job CIUDBCR to the hlq.SCIUSAMP data set. To create the DB2 tables, review the following values:

dbid	The data base identifier. We changed it to D9E1.
vv	For the PLAN, these are the first two digits of the DB2 version number. We changed it to 91 for our DB2 9.1.
hlq	High-level qualifier of the SYSIN DD, which should be replaced by the higher-level qualifier that is assigned at the install. We changed it to CIU.V2R2M0.
db2hlq	The data set HLQ for DB2 SDSNLOAD. We changed it to DB9E9.
db2runhlq	The data set HLQ for DB2 RUNLIB.LOAD. We changed it to DB9EU.

Example 2-6 shows the jcl that we used to create the DB2 tables.

Example 2-6 CIUDBCR sample job

```
//JOB LIB DD DSN=DB9E9.SDSNLOAD,
//      DISP=SHR
//      DD DSN=DB9EU.RUNLIB.LOAD,
//      DISP=SHR
//DSN1TIAD EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
  RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) -
    LIB('DB9EU.RUNLIB.LOAD')
END
/*
/**
//SYSIN DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUMAIN),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUVER),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICSX),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS2),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS3),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS4),
```

```

//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS5),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS6),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS7),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS8),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICS9),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICSA),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCICSB),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUIB2),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURWEB),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURFILE),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURPROG),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURTRAN),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURTDQ),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURTSQ),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUREXIT),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUTSCTB),
//      DISP=SHR
// * THE CIUIB2V CREATES A VIEW USING SYSIBM TABLES
// * DB2 SYSADM AUTHORITY IS REQUIRED
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUIB2V),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUMQ1),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUIMS),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUAFF),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUAFFV),
//      DISP=SHR

```

```

//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIULMSD),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCSSD),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUCSSV),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUREGTB),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUSTSV),
//      DISP=SHR
//      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURESTX),
//      DISP=SHR
// *      Add in the following step to create index for SYSIBM table
// * DB2 SYSADM AUTHORITY IS REQUIRED
// *      DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUIBM1),
// *      DISP=SHR

```

To create the DB2 tables successfully, review the following values that are specified in member CIUMAIN:

qual The OWNER of the database. We changed it to CICSRS3. We specified CICSRS3 in the DB2 variables section of the CICS IA installation and customization program. CICSRS3 is the TSO user ID that we used to install CICS IA.

vol The required DASD volumes. We do not use a specific volume name; therefore, we specify a name of '*' on the STOGROUP statement.

Note: Review the associate SQL members for the above jobs. They are in hlq.SCIUSQL.OUT. You can adjust the values for PRIQTY and SECQTY on the CREATE TABLESPACE statements.

Example 2-7 on page 23 shows our SCIUSQL member CIUMAIN.

Example 2-7 SCIU SQL member CIU MAIN

```
SET CURRENT SQLID='CICRS3';
DROP DATABASE CIUTBLSP;
COMMIT;
    DROP STOGROUP CIUSGTS;
    DROP STOGROUP CIUSGIX;
COMMIT;
    CREATE STOGROUP CIUSGTS VOLUMES('*') VCAT DB9EU;
    CREATE STOGROUP CIUSGIX VOLUMES('*') VCAT DB9EU;
COMMIT;
CREATE DATABASE CIUTBLSP;
COMMIT;
```

2.3.2 Editing job CIUDBND and SCIU SQL member CIUGRANT

We use customized member CIUDBND to bind the DBRMS. Member CIUDBND was copied to the SCIU SAMP.OUT data set. Review the following values:

dbid	Replace this value with the database ID. We changed it to D9E1.
vv	For the PLAN, these are the first two digits of the DB2 version number. We changed it to 91.
hlq	The data set prefix for the CICS IA product. We changed it to CIU.V2R2M0.
db2hlq	The data set HLQ for the DB2 SDSNLOAD. We changed it to DB9E9.
db2runhlq	The data set HLQ for DB2 RUNLIB.LOAD. We changed it to DB9EU.
own	The authorization ID of the owner of the DB2 plan, in our case CICRS3.
qual	To the implicit qualifier for unqualified names of DB2 tables, views, indexes, and so forth. Set this value to the same value as in the CIU MAIN input in the job CIUDBCR. We changed it to CICRS3.

Example 2-8 shows our CIUDBND job.

Example 2-8 Shortened CIUDBND job sample

```
//JOB LIB DD DSN=DB9E9.SDSNLOAD,
// DISP=SHR
// DD DSN=DB9EU.RUNLIB.LOAD,
// DISP=SHR
//BIND EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
```

```

//DBRMLIB DD DSN=CIU.V2R2MO.SCIUDBRM,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
BIND PACKAGE (CPS4) -
  MEMBER (CIUCINB2) -
  OWNER (CICSR3) -
  QUALIFIER (SYSIBM) -
  LIBRARY ('CIU.V2R2MO.SCIUDBRM') -
  SQLERROR (NOPACKAGE) -
  VALIDATE (BIND) -
  ISOLATION (CS) -
  ACTION (REPLACE)
  .....
  .....
  .....
BIND PACKAGE (CPS4) -
  MEMBER (CIULMS) -
  OWNER (CICSR3) -
  QUALIFIER (CICSR3) -
  LIBRARY ('CIU.V2R2MO.SCIUDBRM') -
  SQLERROR (NOPACKAGE) -
  VALIDATE (BIND) -
  ISOLATION (CS) -
  ACTION (REPLACE)
BIND PACKAGE (CPS4) -
  MEMBER (CIUCSS) -
  OWNER (CICSR3) -
QUALIFIER (CICSR3) -
  LIBRARY ('CIU.V2R2MO.SCIUDBRM') -
  SQLERROR (NOPACKAGE) -
  VALIDATE (BIND) -
  ISOLATION (CS) -
  ACTION (REPLACE)
BIND PACKAGE (CPS4) -
  MEMBER (CIUAQRYC) -
  OWNER (CICSR3) -
  QUALIFIER (CICSR3) -
  LIBRARY ('CIU.V2R2MO.SCIUDBRM') -
  SQLERROR (NOPACKAGE) -
  VALIDATE (BIND) -
  ISOLATION (CS) -
  ACTION (REPLACE)
BIND PACKAGE (CPS4) -
  MEMBER (CIUNTSQ2) -
  OWNER (CICSR3) -

```

```

        QUALIFIER (CICRS3) -
        LIBRARY ('CIU.V2R2MO.SCIUDBRM') -
        SQLERROR (NOPACKAGE) -
VALIDATE (BIND) -
        ISOLATION (CS) -
        ACTION (REPLACE)
END
/*
//BIND2 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB DD DSN=CIU.V2R2MO.SCIUDBRM,
// DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
BIND PLAN(CIUBTCH) PKLIST(CPS4.*) -
        OWNER(CICRS3) -
        DYNAMICRULES(BIND) ACT(REP) ISO(CS)
BIND PLAN(CIUCICS) PKLIST(CPS4.*) -
        OWNER(CICRS3) -
        DYNAMICRULES(BIND) ACT(REP) ISO(CS)
END
/*
//*****
//*** GRANT ACCESS TO THE PLANS
//*** EDIT MEMBER CIUGRANT TO GRANT ACCESS TO A RACF USERID OR G
//*** THE RECOMMENDED WAY IS TO USE A RACF GROUP
//*****
//ACCESS EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) -
        LIB('DB9EU.RUNLIB.LOAD')
END
/*
//**
//SYSDSN DD DSN=CIU.V2R2MO.SCIUSQL.OUT(CIUGRANT),
// DISP=SHR

```

2.3.3 Editing SCIUSQL member CIUGRANT

Make sure all users are connected to the new RACF group CIUGROUP. Example 2-9 on page 26 shows our SCIUSQL member CIUGRANT.

Example 2-9 *SCIUSQL member CIUGRANT*

```
-----  
-- Change CICSRS3 to the qualifier of the Db2 table  
-- Should be set to the same value as in CIUMAIN  
--  
-- Change _racfgrp_ To The RACF Group that requires access  
-----  
SET CURRENT SQLID='CICSRS3';  
  
GRANT EXECUTE ON PLAN CIUCICS TO CICS;  
  
GRANT EXECUTE ON PLAN CIUBTCH TO CICS;  
  
COMMIT;
```

2.3.4 Creating the default and IVP application

It is mandatory that you create default and IVP applications. To create default SQL queries and an IVP application do the following steps:

1. Review the following jobs:
 - hlq.SCIUSAMP.OUT(CIUANEW)
 - hlq.SCIUSAMP.OUT(CIUALOAD)
 - hlq.SCIUSAMP.OUT(CIUADDESC)
2. Run customized member CIUANEW in the SCIUSAMP.OUT data set first. On successful completion, run member CIUALOAD. On successful completion, run member CIUADDESC.

CIUANEW

Create the new application definition SQL by reviewing and running the customized sample job CIUANEW in the SCIUSAMP.OUT data set:

- ▶ Make sure the correct CICS IA product qualifier is used. We use CIU.V2R2M0.
- ▶ We do not want to create new applications. Therefore we use MEMBER=CIUAPIVP to create the default IVP application. Example 2-10 on page 27 shows the CIUANEW job that we use.

Example 2-10 CIUANEW job sample

```
//NEWAPP PROC
//STEP010 EXEC PGM=IKJEFT1B,
//          DYNAMNBR=50
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//INFILE   DD DSN=CIU.V2R2MO.SCIUDAT1.OUT(&MEMBER),
//          DISP=SHR
//INTEML   DD DSN=CIU.V2R2MO.SCIUDAT2.OUT(CIUATMPL),
//          DISP=SHR
//OUTFILE  DD DSN=CIU.V2R2MO.SCIUDAT2.OUT(&MEMBER),
//          DISP=SHR
//SYSTSIN  DD DSN=CIU.V2R2MO.SCIUCLIS.OUT(CIUANEWAP),
//          DISP=SHR
//          PEND
//*
//APPL1    EXEC NEWAPP, MEMBER=CIUAPIVP
//*APPL2   EXEC NEWAPP, MEMBER=CIUAP_xxx_
```

CIUALOAD

Example 2-11 on page 28 shows the CIUALOAD job that we use.

Example 2-11 CIUALOAD job

```
//DB2LIB JCLLIB ORDER=DB9EU.PROCLIB
//STEP010 EXEC DSNUPROC,
//          SYSTEM=D9E1,
//          UID='CIU',
//          UTPROC=' '
//STEPLIB DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
/** CIUAPNON IS ALWAYS REQUIRED
//DSNUPROC.SYSREC DD DISP=SHR,
//          DSN=CIU.V2R2M0.SCIUDAT2.OUT(CIUAPNON)
//          DD DSN=CIU.V2R2M0.SCIUDAT2.OUT(CIUAPIVP),
//          DISP=SHR
/**          DD DSN=CIU.V2R2M0.SCIUDAT2.OUT(CIUAP_XXX_),
/**          DISP=SHR
//*****
/** FOR USER DEFINED APPLICATION DEFINITIONS, THE MEMBERS GENERATED
/** BY THE JOB CIUANEW MUST BE CONCATENATED ABOVE. REMOVE THE
/** ASTERISK AND REPLACE "_xxx_" WITH THE APPLICATION CODE.
//*****
//DSNUPROC.SORTWK01 DD DSN=&&TEMP1,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SORTWK02 DD DSN=&&TEMP2,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SORTWK03 DD DSN=&&TEMP3,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SORTWK04 DD DSN=&&TEMP4,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SYSUT1 DD DSN=&&TEMP5,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SORTOUT DD DSN=&&TEMP6,DISP=(NEW,DELETE,DELETE),
//          SPACE=(16384,(20,20),,,ROUND),UNIT=SYSDA
//DSNUPROC.SYSERR DD SYSOUT=*
//DSNUPROC.SYSIN DD *
LOAD DATA
  REPLACE
  RESUME NO
  LOG YES
  INTO TABLE CICSRS3.CIU_SQL_DATA
  WHEN (1:1) = X'40'
          (KEY1          POSITION ( 2: 3) DECIMAL EXTERNAL,
          KEY2          POSITION ( 5: 6) DECIMAL EXTERNAL,
          KEY3          POSITION ( 8: 8) DECIMAL EXTERNAL,
          KEY4          POSITION (10:10) DECIMAL EXTERNAL,
KEY5          POSITION (12:12) DECIMAL EXTERNAL,
          APP          POSITION (14:16) CHAR,
```

SEQ POSITION (18:19) DECIMAL EXTERNAL,
SQL_TEXT POSITION (21:100) CHAR)

/*

CIUADESC

Before running job CIUADESC, review the following variables:

dbid Specifies the DB2 SSID. We use D9E1.
vv Specifies the DB2 version number. We use 91.
hlq Specifies the CICS IA product qualifier. We use CIU.V2R2M0.
db2hlq Specifies the qualifier for the SDSNLOAD data set. We use DB9E9.
db2runhlq Specifies the qualifier for the DB2 RUNLIB.LOAD data set. We use DB9EU.

Example 2-12 shows the CIUADESC job that we use.

Example 2-12 CIUADESC

```
//JOB LIB DD DSN=DB9E9.SDSNLOAD,
// DISP=SHR
// DD DSN=DB9EU.RUNLIB.LOAD,
// DISP=SHR
//DSN1TIAD EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSIN DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUAPDEL),
// DISP=SHR
// DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUADIVP),
// DISP=SHR
//*****
/* FOR USER DEFINED APPLICATION DEFINITIONS, THE MEMBERS CREATED
/* IN DATASET SCIUSQL MUST BE CONCATENATED ABOVE. REMOVE THE
/* '*REM' AND REPLACE "_xxx_" WITH THE APPLICATION CODE.
//*****
/*REM DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUAD_xxx_),
/*REM DISP=SHR
//SYSTSIN DD *
DSN SYSTEM(D9E1)
RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) -
LIB('DB9EU.RUNLIB.LOAD')
END
```

2.4 Running the installation verification programs

You must check the following items to verify that the installation was done correctly:

- ▶ The transaction CIUV, which verifies that all CICS RDO resources are correctly defined and available
- ▶ The CIUIVPLD job, which uploads sample interdependency data to the database, so that you can use the query interface to view it

2.4.1 Running the CIUV transaction

This installation verification step checks CICS RDO definitions to ensure that all software elements (programs, maps, transactions, files, TD-Queues, and DB2 entries) are correctly defined and available.

Clear the panel and run the CIUV transaction, which should end with this message:

```
CIU1002I  INSTALLATION VERIFICATION ENDED SUCCESSFULLY
```

2.4.2 Running the CIUIVPLD job

When the IVP confirms that CICS IA is installed correctly, to familiarize yourself with the product and to check that it is indeed working correctly, you can load some sample interdependency data into the Dependency database objects and use the Query interface to view it.

Edit and run the CIUIVPLD job to load the sample IVP data. The CIUIVPLD job is put into the hlq.SCIUSAMP library by the CICS IA installation procedure. Example 2-13 on page 31 shows our CIUIVPLD job.

Example 2-13 CIUIVPLD job sample

```
//*-----  
/*      REFORMAT INPUT FILE  
/*-----  
//STEP020 EXEC PGM=CIUU040  
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,  
//        DISP=SHR  
//        DD DSN=CIU.V2R2M0.SCIULODE,  
//        DISP=SHR  
//SYSPRINT DD SYSOUT=*  
//INPUT   DD DSN=CIU.V2R2M0.SCIUDAT3(CIUIVPC),  
//        DISP=SHR  
//INPUT2  DD DUMMY  
//OUTPUT  DD DSN=&&DATA2,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),  
//        UNIT=SYSDA,DCB=(RECFM=FB,LRECL=384,BLKSIZE=38400)  
/*-----  
/*      SORT THE INPUT FILE  
/*-----  
//STEP030 EXEC PGM=SORT,COND=(0,NE,STEP020)  
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR  
//SYSUDUMP DD SYSOUT=*  
//SYSOUT  DD SYSOUT=*  
//SORTIN  DD DSN=&&DATA2,DISP=(OLD,DELETE)  
//SORTOUT DD DSN=&&DATA3,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),  
//        UNIT=SYSDA,DCB=*.SORTIN  
//SYSIN   DD *  
        SORT FIELDS=(1,8,A,13,4,A,17,8,A,41,255,A),  
        FORMAT=CH  
        RECORD TYPE=F,LENGTH=(376)  
/*  
/*-----  
/*      RUN THE BATCH PROGRAM CIUU050  
/*-----  
//STEP040 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP020),  
//        DYNAMNBR=20,  
//        PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU055)',  
//        'PLAN(CIUBTCH)','PARM(NOPARM)') <-- NO TimestMP UPD  
/*        'PLAN(CIUBTCH)','PARM(UPD)') <-- TimestMP UPDATE  
/*-----  
/*      IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED  
/*      TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)  
/*-----  
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,  
//        DISP=SHR  
//        DD DSN=CIU.V2R2M0.SCIULODE,  
//        DISP=SHR
```

```

//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2M0.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUINT1 DD DSN=&&DATA3,DISP=(OLD,DELETE)
//*-----
/**          REFRESH CIU_CICS_CHAIN                               *
/**-----
//STEP050 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP020),
//          DYNAMNBR=20,
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU100)',
//          'PLAN(CIUBTCH)')
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2M0.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2M0.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//*-----
/**          REFRESH CIU_CICS_CHAINP
/**-----
/** WARNING - THIS STEP CAN BE COU INTENSIVE
/** THE TABLE IS NO LONGER USED BY CICS IA
/** IF YOU WISH TO RUN IT THEN UNCOMMENT THE STEP
/**-----
/**STEP051 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP020),
/**          DYNAMNBR=20,
/**          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU200)',
/**          'PLAN(CIUBTCH)')
/**STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,
/**          DISP=SHR
/**          DD DSN=CIU.V2R2M0.SCIULODE,
/**          DISP=SHR

```

```
//*      DD DSN=DB9E9.SDSNLOAD,
//*      DISP=SHR
//*SYSPROC DD DSN=CIU.V2R2M0.SCIUCLIS.OUT,
//*      DISP=SHR
//*SYSUDUMP DD SYSOUT=*
//*SYSTSIN DD DUMMY
//*SYSTSPRT DD SYSOUT=*
//*SYSABOUT DD SYSOUT=*
//*SYSOUT DD SYSOUT=*
```

We changed the following parameter in the SCIUCLIS member CIUDB2BT:

hlq The HLQ for CIU product. We changed it to CIU.V2R2M0.

2.5 Updating the Dependency database

The Dependency database contains accumulated data about your applications and the resources that they use. You must regularly update this database to add new information that the Collector records in the VSAM Dependency files. CICS IA provides the following suite of batch jobs to update the database from the VSAM files:

- ▶ CIURES LD Updates the CIU_RESOURCE table used by the Explorer.
- ▶ CIUUPDB1 Updates the CICS table, CIU_CICS_DATA, from the CICS dependency data files, CIUINT1, CIUINT5 and CIUINT6.
- ▶ CIUUPDB2 Updates the DB2 table, CIU_DB2_DATA, from the DB2 dependency data file, CIUINT2.
- ▶ CIUUPDB3 Updates the MQ table, CIU_MQ_DATA, from the MQ dependency data file, CIUINT3.
- ▶ CIUUPDB4 Updates the IMS table, CIU_IMS_DATA, from the IMS dependency data file, CIUINT4.
- ▶ CIUUPDB Updates all tables (CICS, DB2, WMQ, and IMS), from all the dependency data files, CIUINT1 through CIUINT6.

If your Dependency data files are shared by multiple CICS regions, you can use a single run of this suite of jobs to store the Dependency information for all of the regions into the Dependency database.

We decided to run member CIUUPDB to update all tables from all of the Dependency data files. A customized version of the jcl was copied to data set SCIUSAMP.OUT. We recommend that you review the following system variables

to meet your environment requirements. All variables that are used in member CIUUPDB are defined correctly already if the installation and customization program executed successfully.

Example 2-14 through Example 2-33 on page 46 show the relevant jcl snippets for the CICS section, DB2 section, WMQ section, and IMS section of the JCL.

Example 2-14 shows the first part of the CICS section of the JCL.

Example 2-14 CIUUPDB job - part 1

```

/*****
/*-----
/*          RUN THE BATCH PROGRAM CIUUREG
/*-----
//STEP000 EXEC PGM=IKJEFT1B,
//          DYNAMNBR=20,
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUUREG)',
//          'PLAN(CIUBTCH)')
/*-----
/*          IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED
/*          TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)
/*-----
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUCNTL DD DSN=CICSSYSF.CICSTS32.CIUCNTL,
//          DISP=SHR

```

CICS section of job CIUUPDB

Example 2-15 on page 35 shows how this step is executing on IDCAMS REPRO to copy the CICS records that the Collector created to a temporary QSAM file called &&DATA1. We run program CIUUREG to update the CIU_REGION_INFO table.

Example 2-15 CIUUPDB job - part 2

```
//*****  
//*-----  
//*          CONVERT COLLECTED DATA TO QSAM FILE  
//*-----  
//STEP010 EXEC PGM=IDCAMS  
//SYSPRINT DD  SYSOUT=*  
//IN          DD  DSN=CICSSYSF.CICSTS32.CIUINT1,  
//            DISP=SHR  
//OUT         DD  DSN=&&DATA1,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),  
//            UNIT=SYSDA,DCB=(RECFM=VB,LRECL=131,BLKSIZE=13100)  
//SYSIN       DD  *  
              REPRO IFILE(IN),OFIL(OUT)  
//*****  
//*-----  
//*          CONVERT COLLECTED DATA TO QSAM FILE - LONG FILE  
//*-----  
//STEP015 EXEC PGM=IDCAMS  
//SYSPRINT DD  SYSOUT=*  
//IN          DD  DSN=CICSSYSF.CICSTS32.CIUINT5,  
//            DISP=SHR  
//OUT         DD  DSN=&&DATA0,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),  
//            UNIT=SYSDA,DCB=(RECFM=VB,LRECL=361,BLKSIZE=36100)  
//SYSIN       DD  *  
              REPRO IFILE(IN),OFIL(OUT)
```

Example 2-16 shows how we run program CIUU040 to convert the hexadecimal function values that are collected from the argument zero fields of EXEC CICS commands into displayable English words. The output is written to a temporary data set called &&DATA2. A return code of 4 for step STEP020 indicates that there is no CICS data to load. In this case steps STEP030 to STEP051 are skipped.

Example 2-16 CIUUPDB job - part 3

```
//*-----  
//*          REFORMAT INPUT FILE  
//*-----  
//STEP020 EXEC PGM=CIUU040  
//STEPLIB DD  DSN=CIU.V2R2MO.SCIULOAD,  
//            DISP=SHR  
//          DD  DSN=CIU.V2R2MO.SCIULODE,  
//            DISP=SHR  
//SYSPRINT DD  SYSOUT=*  
//INPUT DD  DSN=&&DATA1,DISP=(OLD,DELETE)  
//INPUT2 DD  DSN=&&DATA0,DISP=(OLD,DELETE)  
//OUTPUT DD  DSN=&&DATA2,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
```

```
// UNIT=SYSDA,DCB=(RECFM=FB,LRECL=386,BLKSIZE=38600)
```

Example 2-17 shows the step that sorts the &&DATA2 into an order that matches the indexes set up in the table, to make the next step run more efficiently. The data is sorted into a temporary data set called &&DATA3.

Example 2-17 CIUUPDB job - part 4

```
/*-----  
/*          SORT THE INPUT FILE  
/*-----  
//STEP030 EXEC PGM=SORT,COND=(0,NE,STEP020)  
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR  
//SYSUDUMP DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SORTIN DD DSN=&&DATA2,DISP=(OLD,DELETE)  
//SORTOUT DD DSN=&&DATA3,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),  
//          UNIT=SYSDA,DCB=*.SORTIN  
//SYSIN DD *  
SORT FIELDS=(1,8,A,13,4,A,17,8,A,41,255,A),  
FORMAT=CH  
RECORD TYPE=F,LENGTH=(386)  
/*
```

Example 2-18 shows how we run program CIUU055 to update the CIU_CICS_DATA table. If your version of DB2 is 7 or lower you must run program CIUU050 or program CIUU055 if it is 8 or above. The data that has been collected is cumulative, so that the program has to read all the records, checking to see whether they already exist in the table. Only new records are added to the table. The data is read from &&DATA3.

Example 2-18 CIUUPDB job - part 5

```
/*-----  
/*          RUN THE BATCH PROGRAM CIUU055  
/*-----  
//STEP040 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP020),  
//          DYNAMNBR=20,  
//          PARM=(' %CIUDB2BT ', 'SYS(D9E1)', 'PROG(CIUU055)',  
//          'PLAN(CIUBTCH)', 'PARM(NOPARM)') <-- NO TIMESTMP UPD  
/**          'PLAN(CIUBTCH)', 'PARM(UPD)') <-- TIMESTAMP UPDATE  
/*-----  
/**          IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED  
/**          TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)  
/*-----  
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,  
//          DISP=SHR
```

```

//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUINT1 DD DSN=&&DATA3,DISP=(OLD,DELETE)
//*
//*-----
//*          RUN THE BATCH PROGRAM CIUU056
//*          Update Tables with Detailed CICS Info
//*-----
//STEPO45 EXEC PGM=IKJEFT1B,COND=((0,NE,STEP020),(0,NE,STEP040)),
//          DYNAMNBR=20,
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU056)',
//          'PLAN(CIUBTCH)','PARM(NOPARM)') <-- NO TIMESTAMP UPD
//*          'PLAN(CIUBTCH)','PARM(UPD)') <-- TIMESTAMP UPDATE
//*-----
//*          IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED
//*          TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)
//*-----
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUINT6 DD DSN=CICSSYSF.CICSTS32.CIUINT6,
//          DISP=SHR
//*

```

Choose option UPD if you want to keep the “last used date” current in the data space.

Example 2-19 shows how we run program CIUU100 to update table CIU_CICS_CHAIN using the updated version of CIU_CICS_DATA.

Example 2-19 CIUUPDB job - part 6

```
/*-----  
/*          REFRESH CIU_CICS_CHAIN          *  
/*-----  
//STEP050 EXEC PGM=IKJEFT1B,COND=((0,NE,STEP020),(0,NE,STEP040)),  
//          DYNAMNBR=20,  
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU100)'  
//          'PLAN(CIUBTCH)')  
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,  
//          DISP=SHR  
//          DD DSN=CIU.V2R2MO.SCIULODE,  
//          DISP=SHR  
//          DD DSN=DB9E9.SDSNLOAD,  
//          DISP=SHR  
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,  
//          DISP=SHR  
//SYSUDUMP DD SYSOUT=*  
//SYSTSIN DD DUMMY  
//SYSTSPRT DD SYSOUT=*  
//SYSABOUT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
/*-----  
/*          REFRESH CIU_CICS_CHAINP  
/*-----  
/* WARNING - THIS STEP CAN BE CPU INTENSIVE  
/* THE TABLE IS NO LONGER USED BY CICS IA  
/* IF YOU WISH TO RUN IT THEN UNCOMMENT THE STEP  
/*-----  
/*STEP051 EXEC PGM=IKJEFT1B,COND=((0,NE,STEP020),(0,NE,STEP040)),  
/*          DYNAMNBR=20,  
/*          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU200)'  
/*          'PLAN(CIUBTCH)')  
/*STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,  
/*          DISP=SHR  
/*          DD DSN=CIU.V2R2MO.SCIULODE,  
/*          DISP=SHR  
/*          DD DSN=DB9E9.SDSNLOAD,  
/*          DISP=SHR  
/*SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,  
/*          DISP=SHR  
/*SYSUDUMP DD SYSOUT=*
```



```

/*SYSTSIN DD DUMMY
/*SYSTSPRT DD SYSOUT=*
/*SYSABOUT DD SYSOUT=*
/*SYSOUT DD SYSOUT=*

```

Example 2-20 shows how we convert collected data to a QSAM file.

Example 2-20 CIUUPDB job - part 7

```

/*-----
/*          CONVERT COLLECTED DATA TO QSAM FILE
/*-----
//STEP060 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//IN       DD DSN=CICSSYSF.CICSTS32.CIUINT2,
//          DISP=SHR
//OUT      DD DSN=&&DATA4,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=(RECFM=VB,LRECL=143,BLKSIZE=14300)
//SYSIN    DD *
          REPRO IFILE(IN),OFILE(OUT)

```

DB2 section of job CIUUPDB

Run this step using IDCAMS REPRO to copy the DB2 records, which are located in the DB2 dependency data file that the Collector created, to a temporary QSAM file called &&DATA4. See Example 2-20.

Run program CIUU042 to convert the hexadecimal function values that are collected from EXEC SQL commands into displayable English words. The output is written to a temporary data set called &&DATA5. A return code of 4 for step STEP070 indicates that there is no DB2 data to load. In this case, steps STEP080 to STEP090 are skipped. See Example 2-21.

Example 2-21 CIUUPDB job - part 8

```

/*-----
/*          REFORMAT INPUT FILE
/*-----
//STEP070 EXEC PGM=CIUU042
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//INPUT   DD DSN=&&DATA4,DISP=(OLD,DELETE)
//OUTPUT  DD DSN=&&DATA5,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=(RECFM=FB,LRECL=189,BLKSIZE=18900)

```

Run this step, Example 2-22, to sort &&DATA5 into an order that matches the indexes set up in the table, to make the next step run more efficiently. The data is sorted into a temporary data set called &&DATA6.

Example 2-22 CIUUPDB job - part 9

```

/*-----
/*          SORT THE INPUT FILE
/*-----
//STEP080 EXEC PGM=SORT,COND=(0,NE,STEP070)
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SORTIN DD DSN=&&DATA5,DISP=(OLD,DELETE)
//SORTOUT DD DSN=&&DATA6,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=*.SORTIN
//SYSIN DD *
SORT FIELDS=(1,8,A,17,4,A,21,8,A,37,10,A,47,40,A),
FORMAT=CH
RECORD TYPE=F,LENGTH=(189)
/*

```

Run program CIUU052 to update the CIU_DB2_DATA table, as shown in Example 2-23. The data that was collected is cumulative; therefore, the program has to read all of the records, checking to see whether they already exist in the table. Only new records are added to the table. The data is read from &&DATA6.

Example 2-23 CIUUPDB job - part 10

```

/*-----
/*          RUN THE BATCH PROGRAM CIUU052
/*-----
//STEP090 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP070),
//          DYNAMNBR=20,
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU052)',
//          'PLAN(CIUBTCH)','PARM(NOPARM)') <-- NO TIMESTAMP UPD
/*          'PLAN(CIUBTCH)','PARM(UPD)') <-- TIMESTAMP UPDATE
/*-----
/*          IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED
/*          TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)
/*-----
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,

```

```
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//CIUINT1  DD DSN=&&DATA6,DISP=(OLD,DELETE)
```

MQ Section of job CIUUPDB

Run this step, Example 2-24, using IDCAMS REPRO to copy the MQ records, which are located in the MQ dependency data file that the Collector created, to a temporary QSAM file called &&DATA7.

Example 2-24 CIUUPDB job - part 11

```
//*-----
//*          CONVERT COLLECTED DATA TO QSAM FILE
//*-----
//STEP100 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//IN       DD DSN=CICSSYSF.CICSTS32.CIUINT3,
//          DISP=SHR
//OUT      DD DSN=&&DATA7,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=(RECFM=VB,LRECL=136,BLKSIZE=13600)
//SYSIN    DD *
REPRO IFILE(IN),OFILE(OUT)
```

Example 2-25 shows how to run program CIUU041 to convert the hexadecimal function values that are collected from WMQ commands into displayable English words. The output is written to a temporary data set called &&DATA8. A return code of 4 for STEP110 indicates that there is no WMQ data to load. In this case steps STEP120 to STEP130 are skipped.

Example 2-25 CIUUPDB job - part 12

```
//*-----
//*          REFORMAT INPUT FILE
//*-----
//STEP110 EXEC PGM=CIUU041
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//INPUT    DD DSN=&&DATA7,DISP=(OLD,DELETE)
```

```
//OUTPUT DD DSN=&&DATA8,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
// UNIT=SYSDA,DCB=(RECFM=FB,LRECL=165,BLKSIZE=16500)
```

Run this step, Example 2-26, to sort &&DATA8 into an order that matches the indexes that are set up in the table, which makes the next step run more efficiently. The data is sorted into a temporary data set called &&DATA9.

Example 2-26 CIUUPDB job - part 13

```
/*-----
/*          SORT THE INPUT FILE
/*-----
//STEP120 EXEC PGM=SORT,COND=(0,NE,STEP110)
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SORTIN DD DSN=&&DATA8,DISP=(OLD,DELETE)
//SORTOUT DD DSN=&&DATA9,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=*.SORTIN
//SYSIN DD *
SORT FIELDS=(1,8,A,9,4,A,17,8,A,25,8,A,41,48,A),
FORMAT=CH
RECORD TYPE=F,LENGTH=(165)
/*
```

Run program CIUU051 to update the CIU_MQ_DATA table, as shown in Example 2-27. The data that was collected is cumulative; therefore, the program has to read all of the records, checking to see whether they already exist in the table. Only new records are added to the table. The data is read from &&DATA9.

Example 2-27 CIUUPDB job - part 14

```
/*-----
/*          RUN THE BATCH PROGRAM CIUU051
/*-----
//STEP130 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP110),
//          DYNAMNBR=20,
//          PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUU051)',
//          'PLAN(CIUBTCH)','PARM(NOPARM)') <-- NO TIMESTAMP UPD
/*          'PLAN(CIUBTCH)','PARM(UPD)') <-- TIMESTAMP UPDATE
/*-----
/*          IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED
/*          TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)
/*-----
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
```

```

//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUINT1 DD DSN=&&DATA9,DISP=(OLD,DELETE)

```

IMS section of job CIUUPDB

Run this step, Example 2-28, using IDCAMS REPRO to copy the IMS records, which are located in the IMS dependency data file that the Collector created, to a temporary QSAM file called &&DATAA.

Example 2-28 CIUUPDB job - part 15

```

/*-----
/*          CONVERT COLLECTED DATA TO QSAM FILE
/*-----
//STEP140 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//IN          DD DSN=CICSSYSF.CICSTS32.CIUINT4,
//          DISP=SHR
//OUT         DD DSN=&&DATAA,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=(RECFM=VB,LRECL=98,BLKSIZE=9800)
//SYSIN      DD *
              REPRO IFILE(IN),OFILE(OUT)

```

Run program CIUU043 to convert the hexadecimal function values that are collected from IMS commands into displayable English words, as shown in Example 2-29. The output is written to a temporary data set called &&DATAA. A return code of 4 for STEP150 indicates that there is no IMS data to load. In this case steps STEP160 to STEP170 are skipped.

Example 2-29 CIUUPDB job - part 16

```

/*-----
/*          REFORMAT INPUT FILE
/*-----
//STEP150 EXEC PGM=CIUU043
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,

```

```

//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//SYSPRINT DD  SYSOUT=*
//INPUT    DD DSN=&&DATAA,DISP=(OLD,DELETE)
//OUTPUT   DD DSN=&&DATAB,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=(RECFM=FB,LRECL=131,BLKSIZE=13100)

```

Run this step, Example 2-30, to sort &&DATAA into an order that matches the indexes that are set up in the table. This action makes the next step run more efficiently. The data is sorted into a temporary data set called &&DATAB.

Example 2-30 CIUUPDB job - part 17

```

/*-----
/*          SORT THE INPUT FILE
/*-----
//STEP160 EXEC PGM=SORT,COND=(0,NE,STEP150)
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SORTIN  DD DSN=&&DATAB,DISP=(OLD,DELETE)
//SORTOUT DD DSN=&&DATAC,DISP=(,PASS),SPACE=(CYL,(5,5),RLSE),
//          UNIT=SYSDA,DCB=*.SORTIN
//SYSIN    DD *
SORT FIELDS=(1,8,A,13,4,A,17,8,A,41,4,A,45,8,A),
FORMAT=CH
RECORD TYPE=F,LENGTH=(131)
/*

```

Run program CIUU053 to update the CIU_MQ_DATA table, as shown in Example 2-31. The data that was collected is cumulative; therefore, the program has to read all of the records and check to see whether the records already exist in the table. Only new records are added to the table. The data is read from &&DATAB.

Example 2-31 CIUUPDB job - part 18

```

/*-----
/*          RUN THE BATCH PROGRAM CIUU053
/*-----
//STEP170 EXEC PGM=IKJEFT1B,COND=(0,NE,STEP150),
//          DYNAMNBR=20,
//          PARM=(' %CIUDB2BT ', 'SYS(D9E1)', 'PROG(CIUU053)',
//          'PLAN(CIUBTCH)', 'PARM(NOPARM)') <-- NO TIMESTAMP UPD
/*          'PLAN(CIUBTCH)', 'PARM(UPD)') <-- TIMESTAMP UPDATE

```

```

/*-----
/*      IF YOU WISH TO UPDATE THE DATABASE WITH THE LAST USED
/*      TIME STAMP FOR EACH DB2 ROW THEN CHOOSE PARM(UPD)
/*-----
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2M0.SCIULODE,
//          DISP=SHR
//          DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//SYSPROC DD DSN=CIU.V2R2M0.SCIUCLIS.OUT,
//          DISP=SHR
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//CIUINT1 DD DSN=&&DATAC,DISP=(OLD,DELETE)

```

Example 2-32 shows how to run the SQL query CIUDPCDB to display the number of rows in updated tables.

Example 2-32 CIUUPDB job - part 19

```

/*-----
/*      RUN SQL QUERIES FOR DB2 TABLES
/*-----
//CIUSPACE EXEC PGM=IKJEFT01
//STEPLIB DD DSN=DB9E9.SDSNLOAD,
//          DISP=SHR
//          DD DSN=DB9EU.RUNLIB.LOAD,
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
  RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) PARM('/ALIGN(LHS) MIXED') -
  LIB('DB9EU.RUNLIB.LOAD')
  END
/*
//SYSIN DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIUUPCDB),
//          DISP=SHR

```

Run SQL file CIURESUP to reload the CIU_RESOURCE table, as shown in Example 2-33 on page 46.

Example 2-33 CIUUPDB job - part 20

```
/*-----  
/*      UPDATE THE CIU_RESOURCE TABLE  
/*-----  
//RESLOAD EXEC PGM=IKJEFT01  
//STEPLIB DD DSN=DB9E9.SDSNLOAD,  
//        DISP=SHR  
//        DD DSN=DB9EU.RUNLIB.LOAD,  
//        DISP=SHR  
//SYSPRINT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSTSIN DD *  
DSN SYSTEM(D9E1)  
  RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) -  
    LIB('DB9EU.RUNLIB.LOAD')  
  END  
/*  
//SYSIN DD DSN=CIU.V2R2M0.SCIUSQL.OUT(CIURESIN),  
//        DISP=SHR
```

2.5.1 Installing the CICS IA Explorer

In this section, we describe the required steps to install the CICS IA Explorer on your workstation.

Before you install the Explorer, ask your system administrator for the address of the host where CICS IA and DB2 are installed. In order to connect the CICS IA Explorer, note the following:

- ▶ If you use a standalone DB2 in your environment, you just need the address of the host where CICS IA and DB2 are installed.
- ▶ If you use CICS IA in a DB2 data sharing environment, you need to obtain the VIPA address of the images that are involved.

We use a DB2 data sharing environment that is installed on z/OS images SC66 and SC53. Therefore we use the `/D TCP/IP,,SYSPLEX,VIPADYN` command to display the VIPA IP address that we use to connect the CICS IA Explorer to DB2, as shown in Example 2-34, which shows the command output.

Example 2-34 Display VIPA IP address

```
D TCPIP,,SYSPLEX,VIPADYN  
EZB260I SYSPLEX CS V1R8 331  
VIPA DYNAMIC DISPLAY FROM TCPIP AT SC66  
IPADDR: 9.12.4.24  
TCPNAME MVSNAME STATUS RANK ADDRESS MASK NETWORK PREFIX DIST
```



```

-----
TCPIP   SC61   ACTIVE   255.255.255.0  9.12.4.0   BOTH
TCPIP   SC62   BACKUP 080                                     DEST
IPADDR: 9.12.4.68 LINKNAME: VIPL090C0444
ORIGIN: VIPADEFINE
TCPNAME MVSNAME STATUS RANK ADDRESS MASK NETWORK PREFIX DIST
-----
TCPIP   SC66   ACTIVE   255.255.252.0  9.12.4.0   BOTH
TCPIP   SC53   BACKUP 255                                     DEST
IPADDR: 9.12.4.74 LINKNAME: VIPL090C044A
ORIGIN: VIPADEFINE
TCPNAME MVSNAME STATUS RANK ADDRESS MASK NETWORK PREFIX DIST
-----
TCPIP   SC66   ACTIVE   255.255.255.0  9.12.4.0   BOTH
TCPIP   SC54   BACKUP 080                                     DEST
IPADDR: 9.12.4.85

```

If you use a standalone DB2 installation, you can use the ISPF shell to issue a NETSTAT HOME command that displays the IP address of the host where CICS IA and DB2 are running.

Before you install the Explorer, you also need to obtain the port on which DB2 is listening. Look at the MSTR job log, which lists the required information. In Example 2-35, we use TCPSPORT 37890 to connect to the Explorer.

Example 2-35 DB2 MSTR job log

```

04.33.56 STC16900 DSNL519I -D9E1 DSNLILNR TCP/IP SERVICES AVAILABLE 901
901 FOR DOMAIN wtsc66.itso.ibm.com AND PORT 37890
04.33.56 STC16900 DSNL004I -D9E1 DDF START COMPLETE 902
902 LOCATION DB9E
902 LU USIBMSC.SCPD9E1
902 GENERICLU -NONE
902 DOMAIN wtsc66.itso.ibm.com
902 TCPSPORT 37890
902 SECPORT 0
902 RESPORT 37891
902 IPNAME -NONE
04.33.56 STC16900 DSN9022I -D9E1 DSNYASCP 'START DB2' NORMAL COMPLETION

```

Installing CICS IA Explorer

The Explorer file CIUEXENU is located in data set hlq.SCIUJAVE. To install the Explorer:

1. On your workstation, use FTP to download the CICS IA file CIUEXENU from the given data set on your host system. Carry out the FTP download in BINARY mode.
2. Rename CIUEXENU to CIUEXENU.zip or a file name of your choice with a .zip extension.
3. Extract the contents of the compressed file to a directory of your choice.
4. To launch the Explorer, locate and double-click the file IAExplorer.exe, which is located in a subdirectory called Eclipse. The Explorer launch panel, Figure 2-2, followed by the Welcome panel, Figure 2-3 on page 49, are displayed.

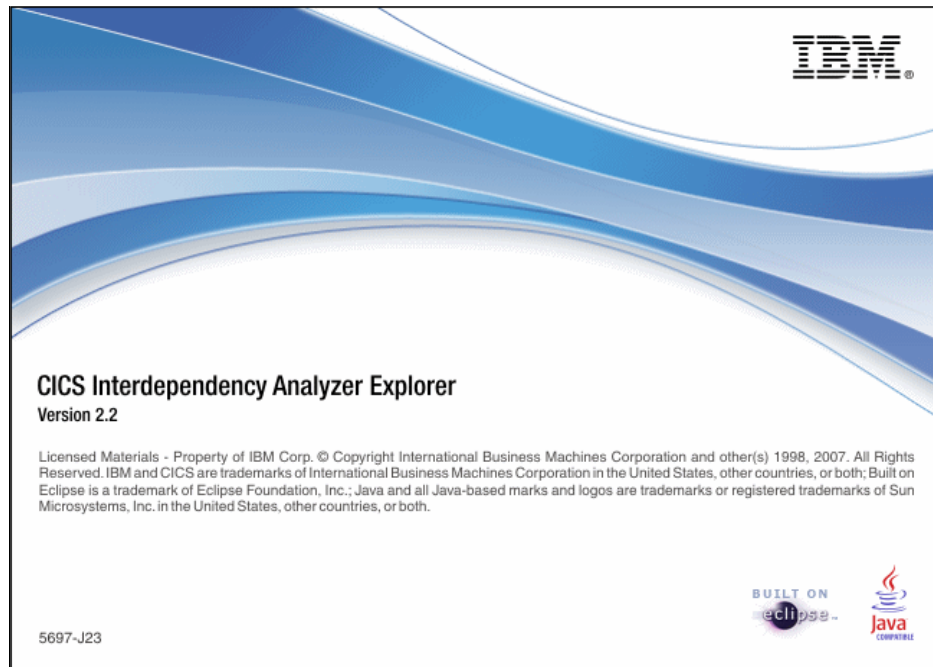


Figure 2-2 Explorer Launch panel

5. To open the connections window, click the green arrow on the Welcome panel, which we show in Figure 2-3 on page 49.

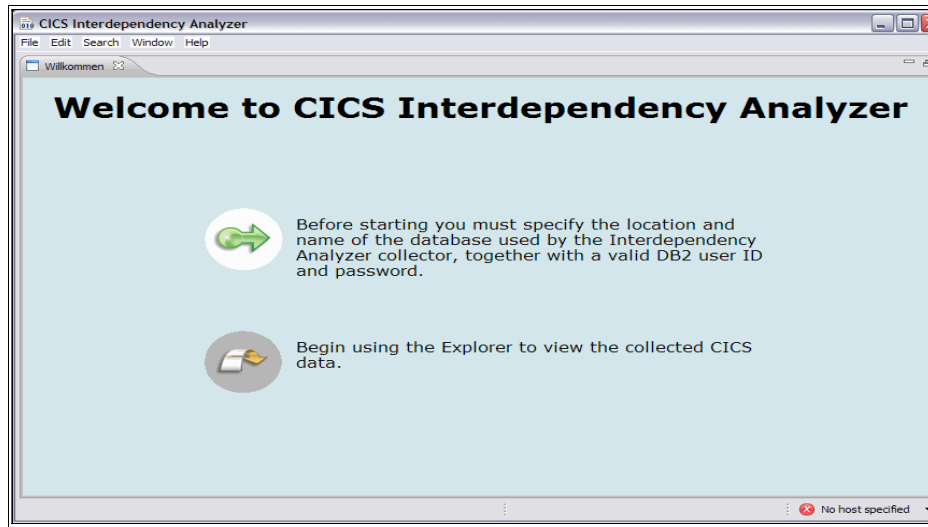


Figure 2-3 CICS IA Explorer Welcome panel

The CICS IA Preferences panel is displayed, as shown in Figure 2-4 on page 50. We enter the information required as follows:

- DB2 location: Contact your DB2 administrator for the information that is required to configure your DB2 location. The required information is located in the MSTR job log that starts the DB2 control address space. We use location DB9E.
- The server address: This is the TCP/IP name or domain name of the z/OS host.
- The TCP/IP port number: The number is 37890. The information required is in the DB2 MSTR job log.
- USERID and password: Enter your user ID and password.
- Schema: We use the qualifier for the CICS IA tables. The qualifier we use is CICSRS3, which was specified in the installation and customization program.

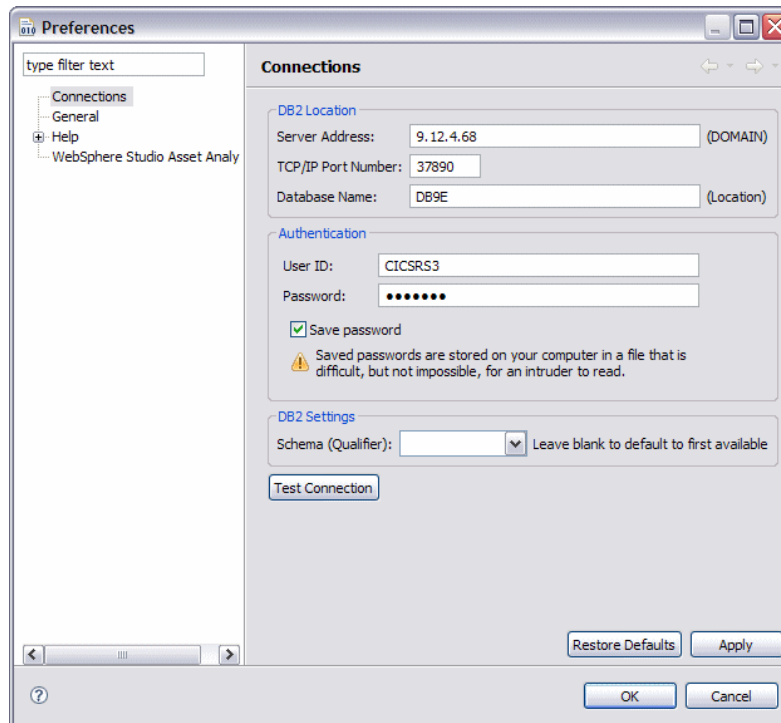


Figure 2-4 CICS IA Explorer Preferences panel

6. Click **Test Connection**. If the Explorer connects successfully, you can click **Apply** and **OK**. After that you can click **Start IA** to begin using the Explorer. See Figure 2-5 on page 51, which shows the CICS IA Explorer panel.

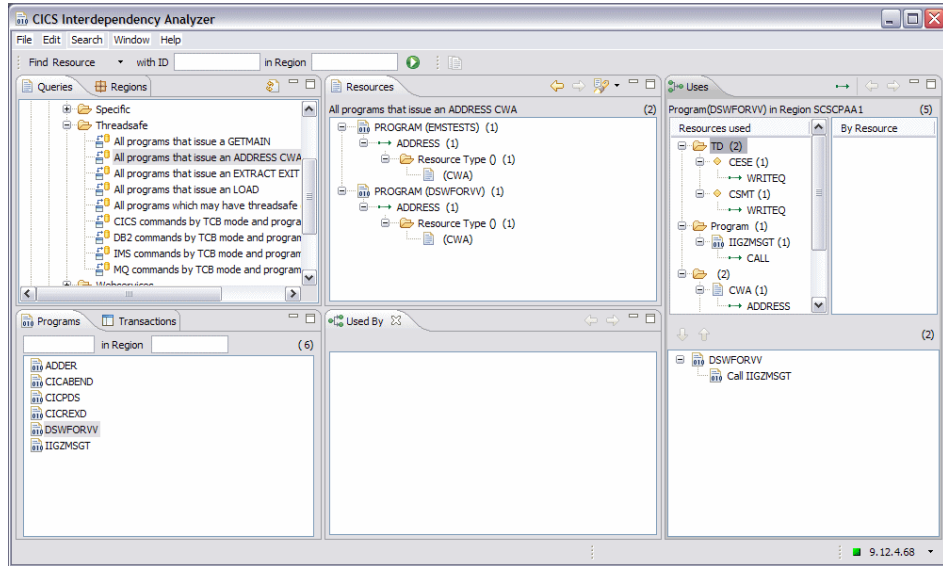


Figure 2-5 CICS IA Explorer panel



Scanner component

In this chapter, we provide information about creating Scanner summary and detailed reports. This information is also in the *CICS IA User Guide and Reference, SC34-6790*. Also, the page references in the following paragraphs refer to this guide.

The *Scanner* scans load modules for instances of program commands that could cause resource dependencies or transaction affinities. The Scanner detects the use of:

- ▶ The dependency-related commands listed in “The Dependency Reporter” on page 8.
- ▶ The affinity-related EXEC CICS API and SPI commands that are listed in Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner.
- ▶ MVS™ POST requests.

You can run the Scanner to produce either a summary report or a detailed report. The recommended way to use the Scanner is to:

1. Create a summary report and module list to identify modules that contain commands that can cause dependencies or affinities.
2. Produce detailed reports to review modules that the summary report identified.

The following jobs are available to create summary and detailed reports. Customized versions of the jobs are copied to the hlq.SCIUSAMP.OUT data set:

- ▶ CIUJCLLS creates a summary report.
- ▶ CIUJCLTS creates a summary report with DB2 output.
- ▶ CIUJCLLD creates a detailed report.
- ▶ CIUJCLTD create a detailed report with DB2 output.

3.1 Creating a summary report

You can create a load module scanner summary report by running the CIUJCLLS job. The job basically provides:

- ▶ A load module scanner summary listing.
- ▶ A data set that contains a list of modules with potential dependencies, affinities, or z/OS post requests.

You can use the list of module names with potential dependencies, affinities, and z/OS post requests as input for the load module scanner detailed report.

Before you run the customized CIUJCLLS jcl, review the following items, as appropriate:

- ▶ The JOB accounting parameters
- ▶ The PARM statement:

```
PARM=' $SUMMARY[,DETAILMODS] '
```

In this statement:

- \$SUMMARY specifies that a summary scan (and report) is required for the entire library, except for CICS modules, CICS tables, and those modules that cannot be loaded (due to an error).
 - DETAILMODS specifies that the names of those modules that contain at least one possible Dependency-causing command are to be written to the sequential file that the INTMOD DD statement defines. You can use this file to restrict a subsequent detailed report by specifying it on the DETAIL DD statement of a detailed report run of the Scanner.
- ▶ The STEPLIB DD statement
Specifies the name of the CICS IA load library that contains the load module scanner program, CIULMS. The default is hlq.SCIULOAD, where *hlq* is the data set qualifier that is assigned during installation.
 - ▶ The INPUT DD statement
Specifies the name of the load library to be scanned.

- ▶ The SYSPRINT DD statement
Specifies the destination for the summary report.
- ▶ The INTMOD DD statement
Specifies the name of the sequential data set where the list of modules with potential resource dependencies is to be sent. You can edit the data set to alter the list of modules to be scanned before you run the Scanner to produce a detailed report.
- ▶ The DETAIL DD statement (dummy)
You do not need this for a summary run.

Figure 3-1 shows an example of the JCL for running the Scanner summary.

```
//SCAN      EXEC PGM=CIULMS,PARM=' $SUMMARY,DETAILMODS '
//STEPLIB  DD DSN=CIU.V2R2MO.SCIULOAD,
//          DISP=SHR
//          DD DSN=CIU.V2R2MO.SCIULODE,
//          DISP=SHR
//INPUT    DD DSN=CICSSYSF.APPL64.LOADLIB,
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//INTMOD   DD DSN=CICSSYSF.CIUOUT1.DETMODS,
//          DISP=(NEW,CATLG,DELETE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000),SPACE=(CYL,(1,1))
//DETAIL   DD DUMMY
//
```

Figure 3-1 CIUJCLLS job to run the Scanner summary report

3.1.1 Contents of a summary report

Figure 3-2 on page 56 shows a sample output from the Scanner summary.

```

CICS INTERDEPENDENCY ANALYZER Version 2.2.0
LOAD MODULE SCANNER - SUMMARY LISTING OF CICSSYSF.APPL64.LOADLIB

```

Module Name	Module Length	Module Language	Language Version	Possible statements.....			Comment
				Affinities	Dependencies	MVS POSTs	
BIGCNT	00000208	ASSEMBLER		1	2	0	
BTSTST	00000258	ASSEMBLER		0	2	0	
COPYGIFS	00000560			0	0	0	
DFHCNV		CICS MOD					
DFHMCT65		CICS TABLE					
DFHPLTI\$		CICS TABLE					
DFHPLTSD		CICS TABLE					
DFHPLTT1		CICS TABLE					
DFHPLTVR		CICS TABLE					
DFHPLT54		CICS TABLE					
DFHOXCFG		CICS MOD					
DFHOXCMO		CICS MOD					
DFHOXSEP		CICS MOD					
DFHOXS3		CICS MOD					
DFHOXVDS		CICS MOD					

Figure 3-2 Scanner summary listing

The Scanner summary contains a separate line that provides the following information about each module in the library:

- ▶ Name
- ▶ Size
- ▶ Language (if determined)
- ▶ Language version: Language Environment or non-Language Environment
- ▶ Number of possible affinity-causing commands
- ▶ Number of possible Dependency-causing commands
- ▶ Number of possible MVS POST commands

If any of the modules show a non zero value in one of the possible statements columns, then they use Dependency-causing commands, affinity-causing commands, or MVS post requests.

Note: The language is determined only if at least one Dependency-causing EXEC CICS command is detected and is derived from the argument zero of the first such command. Therefore, if a load module is created from several source languages, only one language is indicated.

- ▶ The last page of the summary listing shows a statistics summary, as shown in Figure 3-3 on page 57.

```

CICS INTERDEPENDENCY ANALYZER Version 2.2.0
LOAD MODULE SCANNER - SUMMARY LISTING OF CICSSYSF.APPL64.LOADLIB

                          LOAD LIBRARY STATISTICS
=====
Total modules in library           =      22
Total modules scanned              =       8
Total CICS modules/tables (not scanned) =     14
Total modules in error (not scanned) =       0
Total modules containing possible MVS POSTs =       0
Total modules containing possible Dependency commands =       4
Total modules containing possible Affinity commands =       2
  Total ASSEMBLER modules          =       3
  Total C/370 modules              =       0
  Total COBOL modules              =       0
  Total COBOL II modules           =       3
  Total PL/I modules               =       0
Total number of possible Dependency commands =     15
Total number of possible Affinity commands =       3

```

Figure 3-3 Scanner summary listing: Load Library Statistics

The statistics summary includes the total count of:

- Modules in the library
- Modules scanned
- CICS modules and tables (not scanned)
- Modules in error (not scanned)
- Modules that possibly contain Dependency-causing commands
- Modules that possibly contain affinity-causing commands
- Modules that contain possible Dependency-causing commands
- Assembler modules
- C/370™ modules
- COBOL / COBOL II modules
- PL/I modules
- Total number of possible dependency commands
- Total number of possible affinity commands

3.1.2 Creating a summary report with DB2 output

The load module scanner allows you to create a summary report with DB2 data. If you use your own program logic to deal with the output data of the summary report, the CIUJCLTS job updates the CIU_SCAN_SUMMARY DB2 table with the results of the scan. Your own programs can then process the scan results by querying the CIU_SCAN_SUMMARY table.

A customized version of the CIUJCLTS job was copied to the hlq.SCIUSAMP,OUT data set. Review the following parameters before you run the job:

- ▶ The JOB accounting parameters
- ▶ The PARM statement:
PARM=('%CIUDB2BT ', 'SYS(D9E1) ', 'PROG(CIULMS) ', 'PLAN(CIUBTCH) ', ' PARM(' '\$SUMMARY,DETAILMODS,TABLE ' ' '))

In the PARM statement:

- SYS specifies the name of the DB2 subsystem.
- \$SUMMARY specifies that a summary scan (and report) is required for the entire library, except for CICS modules, CICS tables, and those modules that cannot be loaded (due to an error).
- DETAILMODS specifies that the names of those modules that contain at least one possible Dependency-causing command are to be written to the sequential file that is defined by the INTMOD DD statement. This file can restrict a subsequent detailed report by specifying it on the DETAIL DD statement of a detailed report run of the Scanner.
- TABLE specifies that the results of the summary scan are to be written to the DB2 table CIU_SCAN_SUMMARY.
- The STEPLIB DD statement specifies the name of the CICS IA load library in which you installed the Load Module Scanner program, CIULMS. The default is hlq.SCIULOAD, where hlq is the data set qualifier that is assigned during installation.
- The INPUT DD statement specifies the name of the load library to be scanned.
- The SYSPRINT DD statement specifies the destination for the summary report.
- The INTMOD DD statement specifies the name of the sequential data set where the list of modules with potential resource dependencies is to be sent. You can edit the data set to alter the list of modules to be scanned before you run the Scanner to produce a detailed report.
- The DETAIL DD statement (dummy), you do not need this for a summary run.

Example 3-1 on page 59 shows the CIUJCLTS job we used to create a load module scanner summary report.

Example 3-1 CIUJCLTS job

```
//SCAN EXEC PGM=IKJEFT1B,DYNAMNBR=20,
// PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIULMS)',
// 'PLAN(CIUBTCH)','PARM('$SUMMARY,DETAILMODS,TABLE')')
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
// DISP=SHR
// DD DSN=CIU.V2R2MO.SCIULODE,
// DISP=SHR
// DD DSN=DB9E9.SDSNLOAD,
// DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
// DISP=SHR
//INPUT DD DSN=CICSSYSF.APPL65.LOADLIB,
// DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//INTMOD DD DSN=CICSSYSF.CIUOUT2.DETMODS,
// DISP=(NEW,CATLG,DELETE),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000),SPACE=(CYL,(1,1))
//DETAIL DD DUMMY
/*-----
/* RUN SQL QUERIES FOR DB2 TABLES
/*-----
//CIUSPACE EXEC PGM=IKJEFT01
//STEPLIB DD DSN=DB9E9.SDSNLOAD,
// DISP=SHR
// DD DSN=DB9EU.RUNLIB.LOAD,
// DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) PARMS('/ALIGN(LHS) MIXED') -
LIB('DB9EU.RUNLIB.LOAD')
END
/*
//SYSIN DD DSN=CIU.V2R2MO.SCIUSQL.OUT(CIUSPCTS),
// DISP=SHR
//
```

When the job runs successfully, we use SPUFI to issue the select statement shown in Example 3-2 on page 60.

Example 3-2 SQL Select Statement to list table CIU_SCAN_SUMMARY

```

EDIT          CIU.V2R2MO.SCIUSQL.OUT(CIUTEST3) - 01.03           Columns 00001 00080
Command ==>                                         Scroll ==> PAGE
***** ***** Top of Data *****
000001 SELECT * FROM CIU_SCAN_SUMMARY;
***** ***** Bottom of Data *****
```

The select statement we used in Example 3-2 lists all available columns of table CIU_SCAN_SUMMARY. This table stores summary information about every module in the load libraries that are scanned. Example 3-3 shows the truncated output of the SQL select statement. The table contains the following columns:

- ▶ DSNAME
- ▶ PROGRAM
- ▶ LANGUAGE
- ▶ LE
- ▶ CICS/BATCH
- ▶ AFFINITY COUNT
- ▶ MVS POST COUNT
- ▶ DEPENDENCY COUNT

Example 3-3 CIU_SCAN_SUMMARY listing

```

-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
SELECT * FROM CIU_SCAN_SUMMARY;
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
DSNAME                PROGRAM  LANGUAGE  LE    CICS_OR_BATCH  AFFINITY_COUNT  MVS_POST_COUNT
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
CICSSYSF.APPL65.LOADLIB      CICDELAY  ASSEMBLER      CICS      1      0
CICSSYSF.APPL65.LOADLIB      CICDELY2  ASSEMBLER      CICS      1      0
CICSSYSF.APPL65.LOADLIB      ECIPROG   COBOL II      LE    CICS      3      0
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
-
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-
-
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72
DSNE620I NUMBER OF SQL STATEMENTS PROCESSED IS 1
DSNE621I NUMBER OF INPUT RECORDS READ IS 1
DSNE622I NUMBER OF OUTPUT RECORDS WRITTEN IS 19
***** ***** Bottom of Data *****
***** *****
```

3.2 Creating a detailed report

You can request a detailed report from the Scanner by editing and running the job CIUJCLLD, which shows details of the EXEC CICS commands that are used in the scanned load modules.

Change the following statements as appropriate:

- ▶ The PARM statement

```
PARM=' $DETAIL[,ALL] '
```

- \$DETAIL specifies that a detailed scan and report are required. The extent of the scan is defined by either the ALL parameter or the DETAIL DD statement.
- ALL specifies that all modules in the load library are to be scanned for possible Dependency-causing commands.

If ALL is omitted, only those modules that are listed in the file that is specified on the DETAIL DD statement are to be scanned. This file is normally from the INTMOD DD output of a Scanner summary report run, which you can edit before you create a detailed report.

- ▶ The STEPLIB DD statement

Specifies the name of the CICS IA load library in which you installed the Scanner program CIULMS. The default is hlq.SCIULOAD.

- ▶ The INPUT DD statement

Specifies the name of the load library to be scanned.

- ▶ The SYSPRINT DD statement

Specifies the destination for the detailed report.

- ▶ The INTMOD DD dummy statement

You do not need this for a detailed run. Specifies the name of the sequential data set to which the list of modules that contain possible dependency-causing or affinity-causing commands is to be sent. You can edit the data set to alter the list of modules to be scanned before you run the Load Module Scanner to produce a detailed report.

- ▶ The DETAIL DD statement

Specifies the name of the data set that contains the list of modules to be scanned. You can create this list initially as the output from a summary run of the Scanner. If you specify ALL on the PARM statement, change the DETAIL DD statement to specify //DETAIL DD DUMMY.

Figure 3-4 shows a sample JCL for running the Scanner in detail mode.

```
//SCAN      EXEC PGM=CIULMS,PARM=' $DETAIL,ALL '  
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,  
//          DISP=SHR  
//          DD DSN=CIU.V2R2MO.SCIULODE,  
//          DISP=SHR  
//INPUT    DD DSN=CICSSYSF.APPL64.LOADLIB,  
//          DISP=SHR  
//SYSPRINT DD SYSOUT=*  
//DETAIL    DD DSN=CICSSYSF.CIUOUT2.DETMODS,  
//          DISP=SHR  
//INTMOD   DD DUMMY  
//
```

Figure 3-4 Sample JCL for running Scanner in detail mode

3.2.1 Contents of a detailed report

Each detailed report contains a section for each module, with:

- ▶ A header line that gives the name, size, and entry point of the module.
- ▶ A line for each possible Dependency-causing command found, which gives:
 - The offset of the command argument zero declaration from the start of the load module.

Note: This offset is not the same as the offset that the Reporter gives. The offset that the Reporter gives is for the command itself.

- The EDF DEBUG line number, if present, can provide a useful clue for identifying false dependencies. If a section of a load module was translated with the DEBUG option, EDF DEBUG line numbers are given. For such a module, if no DEBUG line number is given, this might indicate that what was found was not an argument zero.
 - What the command appears to be (for example, WRITEQ TS).
- ▶ A summary report of the modules, giving all possible Dependency-causing commands, affinity-causing commands, and MVS POST commands.
- ▶ Library totals, as for the summary report, but only for those modules that are selected for the detailed run.

Figure 3-4 shows an example of a Scanner summary report in detail mode.

Example 3-4 Load module scanner detailed listing

```
CICS INTERDEPENDENCY ANALYZER Version 2.2.0                                04/29/08 Page 1
LOAD MODULE SCANNER - DETAILED LISTING OF CICSSYSF.APPL64.LOADLIB

Module Name - BIGCNT / Load Module Length - 00000208 / Module Entry Point - 00000000
Offset      Storage Content (HEX)                    EDF DEBUG Possible Command           Depcy  Affinity
-----
000001CD 3416700008200000020000000000000000          PUT    CONTAINR           Yes
000001DC 1008A0000800004002                          START  TRANSID           Yes  Trans
Total possible Affinity commands = 1
Total possible Dependency commands = 2
Total possible MVS POSTs = 0

Module Name - BTSTST / Load Module Length - 00000258 / Module Entry Point - 00000000
Offset      Storage Content (HEX)                    EDF DEBUG Possible Command           Depcy  Affinity
-----
00000207 3416F00008A000000000000000000000          PUT    CONTAINR           Yes
00000228 3414F00008A000000000000000000000          GET    CONTAINR           Yes
Total possible Affinity commands = 0
Total possible Dependency commands = 2
Total possible MVS POSTs = 0
```

The last page of the detailed listing displays load library statistics, as shown in Figure 3-5.

```
CICS INTERDEPENDENCY ANALYZER Version 2.2.0
LOAD MODULE SCANNER - DETAILED LISTING OF CICSSYSF.APPL64.LOADLIB

LOAD LIBRARY STATISTICS
=====
Total modules in library = 22
Total modules scanned = 8
Total CICS modules/tables (not scanned) = 14
Total modules in error (not scanned) = 0
Total modules containing possible MVS POSTs = 0
Total modules containing possible Dependency commands = 4
Total modules containing possible Affinity commands = 2
  Total ASSEMBLER modules = 3
  Total C/370 modules = 0
  Total COBOL modules = 0
  Total COBOL II modules = 3
  Total PL/I modules = 0
Total number of possible Dependency commands = 15
Total number of possible Affinity commands = 3
```

Figure 3-5 Scan detail list: final page showing load library statistics

3.2.2 Creating a detailed report with DB2 output

In case you want to create your own program logic to process the result of a detailed report, we recommend that you create a detailed load module scanner

report with DB2 output. You can request a detailed report, with DB2 output, by editing and running the customized CIUJCLTD job, which was copied to the hlq.SCIUSAMP.OUT data set. The CIUJCLTD job updates the CIU_SCAN_DETAIL DB2 table with the results of the scan.

Review and change the following statements as appropriate:

▶ The PARM statement:

```
PARM=('%CIUDB2BT', 'SYS(D9E1)', 'PROG(CIULMS)', 'PLAN(CIUBTCH)', 'PARM(''  
$DETAIL, TABLE''')
```

- SYS specifies the name of the DB2 subsystem.
- TABLE Specifies that the results of the detail scan are to be written to the DB2 table CIU_SCAN_DETAIL.
- \$DETAIL specifies that a detailed scan and report are required. The extent of the scan is defined by either the ALL parameter or the DETAIL DD statement.
- ALL specifies that all modules in the load library are to be scanned for possible Dependency-causing commands.

If ALL is omitted, only those modules that are listed in the file that is specified on the DETAIL DD statement are to be scanned. This file is normally from the INTMOD DD output of a Scanner summary report run, which you can edit before you create a detailed report.

▶ The STEPLIB DD statement

Specifies the name of the CICS IA load library in which you installed the Load Module Scanner program, CIULMS. The default is hlq.SCIULOAD. In the concatenation DD statement, specify the name of the DB2 load library. The default is db2hlq.SDSNLOAD.

▶ The INPUT DD statement

Specifies the name of the load library to be scanned.

▶ The SYSPRINT DD statement

Specifies the destination for the detailed report.

▶ The INTMOD DD dummy statement

You do not need this for a detailed run.

▶ The DETAIL DD statement

Specifies the name of the data set that contains the list of modules to be scanned. This list can be created initially as the output from a summary run of the Scanner. If you specify ALL on the PARM statement, change the DETAIL DD statement to specify //DETAIL DD DUMMY.

Example 3-5 shows the CIUJCLTD job we used to create a detailed report with DB2 output.

Example 3-5 CIUJCLTD

```
//SCAN EXEC PGM=IKJEFT1B,DYNAMNBR=20,
//      PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIULMS)',
//          'PLAN(CIUBTCH)','PARM('$DETAIL,TABLE')')
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//        DISP=SHR
//        DD DSN=CIU.V2R2MO.SCIULODE,
//        DISP=SHR
//        DD DSN=DB9E9.SDSNLOAD,
//        DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//        DISP=SHR
//INPUT DD DSN=CICSSYSF.APPL64.LOADLIB,
//       DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//DETAIL DD DSN=CICSSYSF.CIUOUT2.DETMODS,
//       DISP=(OLD,DELETE)
//INTMOD DD DUMMY
/*-----
/*      RUN SQL QUERIES FOR DB2 TABLES
/*-----
//CIUSPACE EXEC PGM=IKJEFT01
//STEPLIB DD DSN=DB9E9.SDSNLOAD,
//        DISP=SHR
//        DD DSN=DB9EU.RUNLIB.LOAD,
//        DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
RUN PROGRAM(DSNSTEP2) PLAN(DSNSTEP91) PARM('/ALIGN(LHS) MIXED') -
LIB('DB9EU.RUNLIB.LOAD')
END
/*
//SYSIN DD DSN=CIU.V2R2MO.SCIUSQL.OUT(CIUSPCTD),
//      DISP=SHR
```

When the job runs successfully, you can use the **SQL Select** command in Example 3-6 to list all of the available columns of the CIU_SCAN_DETAIL table.

Example 3-6 SQL Select statement to list detailed DB2 table

```

EDIT          CIU.V2R2M0.SCIUSQL.OUT(CIUTEST3) - 01.10          Columns 00001 00080
Command ==>                                         Scroll ==> PAGE
***** ***** Top of Data *****
000001 SELECT * FROM CIU_SCAN_DETAIL;

```

The CIU_SCAN_DETAIL table records detailed information about every command, in specified modules of the load libraries that were scanned, that has the potential to create a resource dependency or a transaction affinity.

Example 3-7 shows the result of the **SQL Select** command. The following columns of the detailed report are listed:

- ▶ DSNAME
- ▶ PROGRAM
- ▶ OFFSET
- ▶ COMMAND
- ▶ RESOURCE TYPE
- ▶ AFFINITY
- ▶ AFFINITY TYPE
- ▶ DEPENDENCY
- ▶ MVS POST
- ▶ COMMAND HEX

Example 3-7 SPUFI output

```

-----+-----+-----+-----+-----+-----+-----+-----+
SELECT * FROM CIU_SCAN_DETAIL;
-----+-----+-----+-----+-----+-----+-----+-----+
DSNAME          PROGRAM      OFFSET  COMMAND  RESOURCE_TYPE  AFFINITY  AFFINITY_TYPE  DEPENDENCY
-----+-----+-----+-----+-----+-----+-----+-----+
CICSSYSF.APPL64.LOADLIB  BIGCNT      461  PUT      CONTAINR      N          Y              Y
CICSSYSF.APPL64.LOADLIB  BIGCNT      476  START    TRANSID      Y          IT              Y
CICSSYSF.APPL64.LOADLIB  BTSTST      519  PUT      CONTAINR      N          Y              Y
CICSSYSF.APPL64.LOADLIB  BTSTST      552  GET      CONTAINR      N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     736  LINK     PROGRAM       N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     753  LINK     PROGRAM       N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     770  LINK     PROGRAM       N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     787  LINK     PROGRAM       N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     855  READ     FILE          N          Y              Y
CICSSYSF.APPL64.LOADLIB  ITSOCMN     889  WRITEQ   TD            N          Y              Y
CICSSYSF.APPL64.LOADLIB  SHOWINFO   506  GET      CONTAINR      N          Y              Y
CICSSYSF.APPL64.LOADLIB  SHOWINFO   594  GETNEXT  CONTAINR      Y          TS              Y
CICSSYSF.APPL64.LOADLIB  SHOWINFO   615  STARTBR  CONTAINR      Y          TS              Y
CICSSYSF.APPL64.LOADLIB  SHOWINFO   678  WRITEQ   TD            N          Y              Y
CICSSYSF.APPL64.LOADLIB  SHOWINFO   712  WRITEQ   TD            N          Y              Y
DSNE610I NUMBER OF ROWS DISPLAYED IS 15
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
-----+-----+-----+-----+-----+-----+-----+-----+
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+-----+-----+-----+
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72

```

3.3 Running the CSECT Scanner

The CSECT Scanner is one of the CICS IA principle components. The CSECT Scanner searches load modules for information to identify the version of each CSECT. Output is stored in DB2 tables and can be used in conjunction with the DB2 dependency tables to identify different program versions.

You can run the CSECT Scanner using the CIUJCLCS job. A customized version of the job was copied to the hlq.SCIUSAMP.OUT data set.

Before you run the CIUJCLCS job, change the following items, as appropriate:

- ▶ The JOB accounting parameters
- ▶ The PARM keyword of the EXEC statement:
PARM='[\$TABLE]' where \$TABLE specifies that the results of the scan are to be added to the DB2 tables CIU_PROGRAM_INFO and CIU_CSECT_INFO
- ▶ The SYS keyword of the EXEC statement
Specifies the name of the DB2 subsystem.
- ▶ The STEPLIB DD statement
Specifies the name of the CICS IA load library where you have installed the CSECT Scanner program, CIUCSS. The default is hlq.SCIULOAS, where hlq is the high-level data set qualifier assigned during installation. In the concatenation DD statement, specify the name of the DB2 load library. The default is db2hlq.SDSNLOAD, where db2hlq is the high-level data set qualifier assigned to the DB2 subsystem during installation.
- ▶ The LOADLIB DD statement
Specifies the name of the load library to be scanned.
- ▶ The SYSPRINT DD statement
Specifies the destination for the printed report.
- ▶ The SYSPROC DD statement.
Specifies the name of the CICS IA CLIST library. The default is hlq.SCIUCLIS, where hlq is the high-level data set qualifier that is assigned during installation.

Example 3-8 on page 68 shows the CIUJCLCS job.

Example 3-8 CIUJCLCS

```
//SCAN EXEC PGM=IKJEFT1B,DYNAMNBR=20,
//      PARM=('%CIUDB2BT','SYS(D9E1)','PROG(CIUCSS)',
//          'PLAN(CIUBTCH)','PARM('$TABLE')')
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,
//        DISP=SHR
//        DD DSN=CIU.V2R2MO.SCIULODE,
//        DISP=SHR
//        DD DSN=DB9E9.SDSNLOAD,
//        DISP=SHR
//SYSPROC DD DSN=CIU.V2R2MO.SCIUCLIS.OUT,
//        DISP=SHR
//LOADLIB DD DSN=CICSSYSF.APPL64.LOADLIB,
//        DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
/*-----
/*      RUN SQL QUERIES FOR DB2 TABLES
/*-----
//CIUSPACE EXEC PGM=IKJEFT01
//STEPLIB DD DSN=DB9E9.SDSNLOAD,
//        DISP=SHR
//        DD DSN=DB9EU.RUNLIB.LOAD,
//        DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(D9E1)
  RUN PROGRAM(DSNTEP2) PLAN(DSNTEP91) PARMS('/ALIGN(LHS) MIXED') -
  LIB('DB9EU.RUNLIB.LOAD')
END
/*
//SYSIN DD DSN=CIU.V2R2MO.SCIUSQL.OUT(CIUSPCCS),
//      DISP=SHR
```

The following output is available when the CIUJCLCS job runs successfully:

- ▶ A printed report, as shown in Example 3-9 on page 69. The printed report has two line formats: one for load module information and an indented one for CSECT information.

- ▶ DB2 table CIU_PROGRAM_INFO contains the load module information, which your own program logic can process. The following columns are available:
 - DSNAME = Data set name
 - PROGRAM = Load module name
 - PROGLen = Load module length in hexadecimal
 - ENTRY_POINT = Entry point offset in hexadecimal
 - ALIAS_OF = If the program name is an alias, this is the program for which it is an alias
 - LINKER_NAME = Identifier of the binder or link-editor
 - LINKER_VERSION = Version number of the binder or link-editor (VV.RR)
 - LINKED = Date and time the program was bound or link-edited
 - AMODE = Addressing mode
 - RMODE = Residence mode

- ▶ DB2 table CIU_CSECT_INFO contains information about CSECTs within the load modules. Your own program logic can use the following columns:
 - DSNAME = Data set name
 - PROGRAM = Load module name
 - PROGLen = Load module length in hexadecimal
 - LINKED = Date and time program was bound or link-edited
 - CSECT_NAME = CSECT name
 - TRAN_1_DATE = First translation date (YYYYDDD)
 - TRAN_1_NAME = First translator identifier
 - TRAN_1_VERSION = First translator version (VV.RR)
 - TRAN_2_DATE = Second translation date (YYYYDDD)
 - TRAN_2_NAME = Second translator identifier
 - TRAN_2_VERSION = Second translator version (VV.RR)
 - USER_DATA_DATE = User data date (YYYYDDD)
 - USER_DATA = User data
 - HMASPZAP_DATE = ZAP date (YYYYDDD)
 - HMASPZAP_DATA = ZAP data

Example 3-9 shows the CSECT scanner output listing.

Example 3-9 CSECT scanner output listing

```

1CICS INTERDEPENDENCY ANALYZER Version 2.2.0                                05/06/08   Page   1
CSECT SCANNER - LISTING OF: CICSYSF.APL64.LOADLIB
0

```

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE		
CSECT	T1date	T1name	T1ver T2date	T2name	T2ver	UsrDate	UserData		ZAPdate	ZAPdata
BIGCNT	00000208	00000000		5695PMB01	01.06	2006145113151	31	ANY		

DFHEAI	2005061	569623400	01.04						
BIGCNT	2006145	569623400	01.05						
DFHEAIO	2005061	569623400	01.04						
BTSTST	00000258	00000000		5695PMB01	01.06	2006145144104	31	ANY	
DFHEAI	2005061	569623400	01.04						
BTSTST	2006145	569623400	01.05						
DFHEAIO	2005061	569623400	01.04						
COPYGIFS	00000560	00000000		5695PMB01	01.06	2006130195029	31	24	
DFH\$115	2006130	569623400	01.05						
DFHCNV	00000600	00000028		5695PMB01	01.08	2007268103322	31	ANY	
DFH\$CNV1	2007268	569623400	01.05						
DFH\$CNV2	2007268	569623400	01.05						
DFH\$CNV3	2007268	569623400	01.05						
FCENTRY	2007268	569623400	01.05						

To verify that we got the correct contents in the DB2 tables, we use SPUFI to issue the **SQL Select** command, as shown in Example 3-10.

Example 3-10 SQL SELECT to list CIU_CSECT_INFO

```
SELECT * FROM CIU_CSECT_INFO;
```

Example 3-12 on page 71 shows the output that we received in response to the **SQL Select** command. The information in the SPUFI output corresponds to the output we received in the printed CSECT Scanner report.

You can use your own logic to interpret the Translator name. DB2 table CIU_TRANSLATORS contains a pre loaded translator name- to description reference that allows you to translate the TRAN_1_NAME column of table CIU_CSECT_INFO. Example 3-11 shows a partial listing of the CIU_TRANSLATORS table.

Example 3-11 Partial listing of CIU_TRANSLATORS table

```
SELECT * FROM CIU_TRANSLATORS;
-----+-----+-----+-----+
TRANSLATOR_NAME DESCRIPTION
-----+-----+-----+-----+
360SAS036      S/360 OS ASSEMBLER (E)
360SAS037      S/360 OS ASSEMBLER (F)
5734AS100      OS ASSEMBLER H
5741SC103      OS/V5 ASSEMBLER (XF)
516896201      ASSEMBLER H V2
566896201      ASSEMBLER H V2
569623400      HIGH-LEVEL ASSEMBLER
360SC0503      S/360 OS COBOL (E)
360SCB524      S/360 OS COBOL (F)
360SCB545      S/360 OS FULL ANS COBOL V1 V2
5734CB100      OS FULL ANS COBOL V3 (ANS3)
5734CB200      OS FULL ANS COBOL V4 (ANS4)
5740CB100      OS/V5 COBOL R2M2 (VSR2)
40CB1          OS/V5 COBOL R2M2 (VSR2)
5740CB103      OS/V5 COBOL R2M3 R2M4 (VSR1)
566895801      VS COBOL II
566895807      COBOL/370 or COBOL for MVS & VM
5688197        COBOL/370 or COBOL for MVS & VM
5648A2500      COBOL for OS/390 and VM V2
565565300      Enterprise COBOL for z/OS V3
360SF0092      S/360 OS FORTRAN IV (E)
```



```

360SF0520      S/360 OS FORTRAN IV (G)
360SF0500      S/360 OS FORTRAN IV (H)
5734-F01        FORTRAN CODE AND GO COMPILER
5734-F02        FORTRAN IV G1
5734-F03        FORTRAN IV H EXTENDED
5799-AAW        FORTRAN IV H EXTENDED PLUS
5748-F03        VS FORTRAN V1
5668-806        VS FORTRAN V2 (COMP/LIB/DEBUG)
5688-087        VS FORTRAN V2 (COMP/LIB)
  5796-PKR      Ext. Exponent Range for FORTRAN
360SNL511      S/360 OS PL/I (F)
5734-PL1        OS PL/I OPTIMIZING COMPILER V1
5734-PL2        OS PL/I CHECKOUT COMPILER
5734-PL3        OS PL/I COMPILER & LIBRARY MVS
5668-909        OS PL/I V2 (COMP/LIB/TEST)
5668-910        OS PL/I V2 (COMP/LIB)
5688-235        PL/I for MVS AND VM V1
5655-B22        PL/I for OS/390 V2
5655-H31        Enterprise PL/I for z/OS V3
5713AAG        C for SYSTEM/370 (MVS)
5688040        C/370 COMPILER V1
5688187        C/370 COMPILER V2
5688216        SAA AD/Cycle C/370
5655121        C/C++ for MVS/ESA
5645001        C/C++ OS/390 R2
5647A01        C/C++ OS/390 R4
5655B85        C/C++ Productivity Tools for OS/390
5694A01        C/C++ z/OS R5
5734-XM6        APL/360
5748-AP1        VS APL
5668-899        APL2 VERSION 1
5688-228        APL2 VERSION 2
5706-292        ADA/370
5709-026        ADA COMPILER MVS V2
TRANSLATOR_NAME DESCRIPTION
-----+-----+-----+-----+-----+-----+-----+-----+-----+
5688-194      AD/Cycle CODE/370 R1
360SAL531     S/360 OS ALGOL (F)
5748-XX1     VS BASIC

```

Therefore the TRAN_1_NAME column in Example 3-12 could be translated to the HIGH-LEVEL Assembler translator.

Example 3-12 SPUFI output

```

-----+-----+-----+-----+-----+-----+-----+-----+-----+
PROGRAM  PROGLEN  LINKED           CSECT_NAME  TRAN_1_DATE  TRAN_1_NAME  TRAN_1_VERSION
-----+-----+-----+-----+-----+-----+-----+-----+-----+
BIGCNT   00000208  2006-05-25-11.31.51.000000  DFHEAI      2005061     569623400    01.04
BIGCNT   00000208  2006-05-25-11.31.51.000000  BIGCNT      2006145     569623400    01.05
BIGCNT   00000208  2006-05-25-11.31.51.000000  DFHEAIO     2005061     569623400    01.04
BTSTST   00000258  2006-05-25-14.41.04.000000  DFHEAI      2005061     569623400    01.04
BTSTST   00000258  2006-05-25-14.41.04.000000  BTSTST      2006145     569623400    01.05
BTSTST   00000258  2006-05-25-14.41.04.000000  DFHEAIO     2005061     569623400    01.04
COPYGIFS 00000560  2006-05-10-19.50.29.000000  DFH$115     2006130     569623400    01.05
DFHCNV   00000600  2007-09-25-10.33.22.000000  DFH$CNV1    2007268     569623400    01.05
DFHCNV   00000600  2007-09-25-10.33.22.000000  DFH$CNV2    2007268     569623400    01.05
DFHCNV   00000600  2007-09-25-10.33.22.000000  DFH$CNV3    2007268     569623400    01.05
DFHCNV   00000600  2007-09-25-10.33.22.000000  FCENTRY     2007268     569623400    01.05
DFHCNV   00000600  2007-09-25-10.33.22.000000  ICENTRY     2007268     569623400    01.05

```

DFHCNV	00000600	2007-09-25-10.33.22.000000	TENTRY	2007268	569623400	01.05
DFHCNV	00000600	2007-09-25-10.33.22.000000	TSENTRY	2007268	569623400	01.05
DFHCNV	00000600	2007-09-25-10.33.22.000000	PCENTRY	2007268	569623400	01.05
DFHMCT65	00000250	2007-10-02-13.50.44.000000	DFHMCT	2007275	569623400	01.05

To verify that table CIU_PROGRAM_INFO was updated with the relevant load module information, we used the **SQL Select** command shown in Example 3-13.

Example 3-13 SQL Select to list table CIU_PROGRAM_INFO

```
SELECT * FROM CIU_PROGRAM_INFO;
```

The **SQL Select** command lists all available columns of table CIU_PROGRAM_INFO. Example 3-14 shows the output that we received in response to the **SQL Select** command.

Example 3-14 SPUFI listing of CIU_PROGRAM_INFO

PROGRAM	PROGLEN	ENTRY_POINT	ALIAS_OF	LINKER_NAME	LINKER_VERSION	LINKED	AMODE	RMODE
BIGCNT	00000208	00000000		5695PMB01	01.06	2006-05-25-11.31.51.000000	31	ANY
BTSTST	00000258	00000000		5695PMB01	01.06	2006-05-25-14.41.04.000000	31	ANY
COPYGIFS	00000560	00000000		5695PMB01	01.06	2006-05-10-19.50.29.000000	31	24
DFHCNV	00000600	00000028		5695PMB01	01.08	2007-09-25-10.33.22.000000	31	ANY
DFHMCT65	00000250	00000000		5695PMB01	01.08	2007-10-02-13.50.44.000000	31	ANY
DFHPLTI\$	00000038	00000000		5695DF108	02.10	2002-01-31-20.50.15.000000	31	ANY
DFHPLTSD	00000038	00000000		5695PMB01	01.08	2007-10-12-11.30.12.000000	31	ANY
DFHPLTT1	00000040	00000000		5695PMB01	01.08	2007-10-12-10.12.35.000000	31	ANY
DFHPLTVR	00000038	00000000		5695PMB01	01.08	2007-12-04-04.26.35.000000	31	ANY
DFHPLT54	00000038	00000000		5695PMB01	01.05	2005-08-09-11.00.20.000000	31	ANY
DFHOXCFG	00002F90	00000020		5695PMB01	01.08	2007-08-30-07.25.25.000000	31	ANY
DFHOXCMO	00001C70	00000020		5695PMB01	01.06	2006-05-03-15.19.02.000000	31	ANY
DFHOXSEP	000018F8	00000020		5695PMB01	01.06	2006-05-05-11.56.42.000000	31	ANY
DFHOXS3	000007F0	00000000		5695PMB01	01.08	2007-08-28-10.27.28.000000	31	24
DFHOXVDS	000022E0	00000020		5695PMB01	01.08	2007-08-30-07.27.20.000000	31	ANY
DFHOXW01	00001EC0	00000020		5695PMB01	01.08	2007-08-29-15.48.57.000000	31	ANY



The Collector

The *Collector* is used in real time to detect transaction resource dependencies in a running CICS region and to save details of the dependencies or affinities in an z/OS data space. This data is subsequently saved to DASD.

4.1 What does CICS IA monitor

Figure 4-1 shows the CICS API commands that are monitored by CICS IA.

```
SEND MAP, RECEIVE MAP
ALLOCATE, CONNECT PROCESS, SEND CONVID/ SESSION, CONVERSE
CONVID/ SESSION, FREE
HANDLE ABEND PROGRAM
READ FILE, WRITE FILE, REWRITE FILE, DELETE FILE, STARTBR FILE,
READNEXT FILE,
READPREV FILE, ENDBR FILE, RESETBR FILE, UNLOCK FILE
START
WAIT JOURNALNUM/NAME, WRITE JOURNALNUM/NAME
DEFINE COUNTER/ DOUNTER, DELETE COUNTER/ DOUNTER, GET
COUNTER/DCOUNTER
QUERY COUNTER/ DOUNTER, REWIND COUNTER/ DOUNTER, UPDATE
COUNTER/ DOUNTER
LINK, LOAD, RETURN TRANSID, XCTL
ENQ, DEQ
READQ TS, WRITEQ TS, DELETEQ TS
READQ TD, WRITEQ TD, DELETEQ TD
WEB ENDBROWSE FORMFIELD/ HTTPHEADER, WEB EXTRACT, WEB READ
FORMFIELD/ HTTPHEADER,
WEB READNEXT FORMFIELD/ HTTPHEADER, WEB RECEIVE, WEB RETRIEVE,
WEB SEND,
WEB STARTBROWSE FORMFIELD/ HTTPHEADER, WEB WRITE HTTPHEADER
ADDRESS CWA, ASSIGN APPLID
FREEMAIN, GETMAIN, GETMAIN SHARED,
POP HANDLE, PUSH HANDLE
SYNCPOINT, SYNCPOINT ROLLBACK
```

Figure 4-1 EXEC CICS API commands monitored by CICS IA

Figure 4-2 on page 75 shows the CICS SPI commands that are monitored by CICS IA.

```

INQUIRE BRFCILITY, SET BRFCILITY
CREATE CORBASERVER, INQUIRE CORBASERVER, SET CORBASERVER,
PERFORM CORBASERVER, DISCARD CORBASERVER
CREATE DB2ENTRY, INQUIRE DB2ENTRY, SET DB2ENTRY, DISCARD
DB2ENTRY
CREATE DJAR, INQUIRE DJAR, PERFORM DJAR, DISCARD DJAR
CREATE FILE, INQUIRE FILE, SET FILE, DISCARD FILE
INQUIRE JOURNALNUM/ NAME, SET JOURNALNUM/ NAME, DISCARD
JOURNALNUM/ NAME
INQUIRE JVMPROFILE
CREATE PROGRAM, INQUIRE PROGRAM, SET PROGRAM, DISCARD PROGRAM
CREATE PCPIPSERVICE, INQUIRE TCPIPSERVICE, SET TCPIPSERVICE,
DISCARD TCPIPSERVICE
CREATE TSMODEL, INQUIRE TSMODEL, DISCARD TSMODEL
INQUIRE TSPool
INQUIRE TSQUEUE, SET TSQUEUE, INQUIRE TSQNAME, SET TSQNAME
CREATE TRANSACTION, INQUIRE TRANSACTION, SET TRANSACTION,
DISCARD TRANSACTION
CREATE TDQUEUE, INQUIRE TDQUEUE, SET TDQUEUE

```

Figure 4-2 EXEC CICS SPI commands monitored by CICS IA

Figure 4-3 shows the CICS FEPI commands that are monitored by CICS IA.

```

ALLOCATE PASSCONVID, ALLOCATE POOL
CONVERSE DATASTREAM CONVID, CONVERSE DATASTREAM POOL,
CONVERSE FORMATTED CONVID, CONVERSE FORMATTED POOL
EXTRACT CONV, EXTRACT FIELD, EXTRACT STSN
FREE
ISSUE
RECEIVE/ SEND DATASTREAM, RECEIVE/ SEND FORMATTED
REQUEST PASSTICKET
START
INQUIRE CONNECTION, SET CONNECTION
INQUIRE NODE, SET NODE
ADD POOL, INSTALL POOL, DELETE POOL,
INQUIRE POOL, SET POOL, DISCARD POOL
INSTALL PROPERTYSET, INQUIRE PROPERTYSET, DISCARD PROPERTYSET
INQUIRE TARGET, SET TARGET

```

Figure 4-3 EXEC CICS FEPI commands monitored by CICS IA

Figure 4-4 on page 76 shows the new resources for CICS TS3.1 and CICS TS3.2 that CICS IA monitors for dependencies.

```
PIPELINES
WEBSERVICES
URIMAPS
DOCTEMPLATES
CONTAINERS
CHANNELS
EXITS
GETMAIN STORAGE
LIBRARY
IPCONN
```

Figure 4-4 New CICS TS resources monitored in CICS IA V2.2

Figure 4-5 shows the non EXEC CICS commands that CICS IA monitors for dependencies.

```
EXEC SQL ALTER, EXEC SQL CLOSE, EXEC SQL CREATE,
EXEC SQL DELETE, EXEC SQL DESCRIBE, EXEC SQL DROP,
EXEC SQL EXECUTE, EXEC SQL EXECUTE IMMEDIATE,
EXEC SQL EXPLAIN, EXEC SQL FETCH, EXEC SQL INSERT,
EXEC SQL OPEN, EXEC SQL PREPARE, EXEC SQL SELECT,
EXEC SQL UPDATE
MQCLOSE, MQGET, MQOPEN, MQPUT, MQPUT1
EXEC DLI
CALL (dynamic COBOL calls)
```

Figure 4-5 Non EXEC CICS commands monitored by CICS IA

Figure 4-6 shows the EXEC CICS commands that CICS IA monitors for InterTransaction affinities.

```
ENQ / DEQ
READQ TS, WRITEQ TS, DELETEQ TS
ADDRESS CWA
LOAD
RELEASE
GETMAIN SHARED
FREEMAIN
RETRIEVE WAIT
DELAY
POST
START
CANCEL
```

Figure 4-6 EXEC CICS commands that are monitored for Inter transaction affinities

Figure 4-7 shows the EXEC CICS commands that CICS IA monitors for transaction-system affinities.

ENABLE / DISABLE PROGRAM
EXTRACT EXIT
INQUIRE
SET
PERFORM
RESYNC
DISCARD
CREATE
WAIT EXTERNAL
WAIT EVENT
WAITCICS
COLLECT STATISTICS

Figure 4-7 EXEC CICS commands that are monitored for transaction-system affinities

4.1.1 Collector components

The Collector consists of:

- ▶ A control transaction, CINT
- ▶ An autosave transaction, CINB
- ▶ A number of global user exit programs
- ▶ A task-related user exit program

The data is collected by the global user exit programs at a number of exit points. For more information about the exit points, refer to the *CICS IA User's Guide*, SC34-6790-00, Chapter 5, Running the collector.

The exit programs intercept the EXEC CICS commands that are needed to deduce dependencies. The exit programs coexists with any other exit programs at the same exit points. They can be placed before or after other exit programs without any of the exit programs being affected. The recommendation is to put the CICS IA exits after any other exit programs at the same point.

4.2 The CINT transaction

You can monitor and control the Collector through the CINT transaction, which enables you to start, pause, continue, and stop the collection of Dependency data into the tables in the data space. Using the CINT transaction, you can also specify which Dependency commands and which transactions data is to be collected. Figure 4-8 on page 78 illustrates the CICS IA Collector.

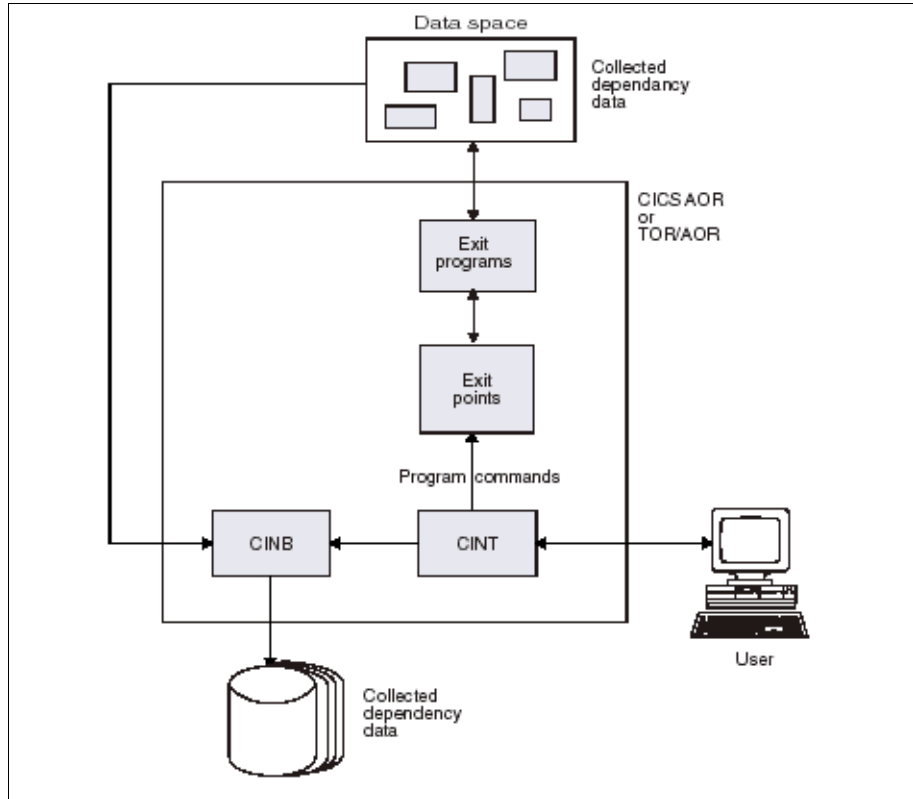


Figure 4-8 The CICS IA Collector

Because the focus of this book is on the use of the other components, we do not include details of controlling the Collector. You can refer to the *CICS IA User's Guide and Reference*, SC34-6365 for more information about how to control the Collector. However, we do show the option menus and explain some of the new features that are added in CICS IA. We also show how to override the default options.

4.2.1 The CINT menu

Figure 4-9 on page 79 shows the initial window that is displayed when CINT is entered.


```
CIU000          CICS Interdependency Analyzer for z/OS - V2R2M0      2008/04/22
                  Main Administration Menu                          09:28:16AM

Select one of the following. Then press Enter.

  1  Operations Menu.
  2  Configure Region Options.
  3  Configure Global Options.

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC34

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

Figure 4-9 CINT transaction main menu

In Figure 4-9, option 3 is used to select the global options, in cases where CICS IA is collecting data on multiple regions. Figure 4-10 on page 80 shows the global options window.

```

CIU300          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                  Global Options Menu                                  12:52:24PM

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

Control options
  VSAM file sharing . . . . . : Y (Yes/No)
  High Level Trace . . . . . : Y (Yes/No)

National Language Option . . . : E Code: ENU

Date and Time Formats
  Date . . . . . 4 1. MMDDYY 2. DDMMYY Separator . . . . . /
                    3. YYMMDD 4. YYYYMMDD

  Time . . . . . 1 1. 12 hrs 2. 24 hrs Separator . . . . . :

CICS Sysid: Z328 CICS Applid: IYDZZ328 TermID: TC34

F1=Help      F2=          F3=End      F4=          F5=Refresh  F6=
F7=          F8=          F9=          F10=         F11=        F12=Cancel

```

Figure 4-10 CINT - Global Options menu

In Figure 4-9 on page 79, if option 2 Configure Region Options is selected from the Main Administration menu, then Figure 4-11 on page 81 is displayed.

```

CIU200          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                  Region Configuration Menu                               12:54:50PM

Type action code then press ENTER.                                     More :
1=Add Region      3=Delete Region   5=DB2/IMS/MQ Options  7=Affinity Options
2=Copy Region    4=CICS Options     6=General Options   8=Time/Date Options

      CICS      CICS  New      New
Act  Applid    Sysid  Applid  Sysid  Status      Collecting
DEFAULTS  DFTS
ALL       ALL
IYDZZ138  Z138                                UNCONNECTED
IYDZZ228  Z228                                UNCONNECTED
IYDZZ238  Z238                                UNCONNECTED
IYDZZ318  Z318                                UNCONNECTED
IYDZZ328  Z328                                STOPPED

CICS Sysid:  Z328  CICS Applid:  IYDZZ328  TermID:  TC34

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

```

Figure 4-11 CINT: Region Configuration menu

In this menu in Figure 4-11, you can configure:

- ▶ The DEFAULT options for all regions
- ▶ Options for ALL regions
- ▶ Options for a single region

The individual CICS region options are initially set to blanks. The DEFAULT options are always set. The CICS IA collector starts up and uses the DEFAULT options unless it is overridden by a REGION option that was set to a non blank.

The option panels are displayed when you select action items 4 to 9. Action items 1, 2, and 3 are used to add, copy, and delete a region.

Note: The STATUS of regions IYDZZ138, IYDZZ228, IYDZZ238, and IYDZZ318 are UNCONNECTED, which indicates that there is no CICS connection setup between these regions or the CICS regions are inactive.

To display and modify the DEFAULT CICS options, press ENTER. See Figure 4-12 on page 82.

```

CIU240          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                  CICS Resources Options for                          01:07:06PM
                  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 . . . . Y Others . . . . Y

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 TCIPService . Y FEPI . . . . Y Journals . . . Y
Library . . . Y IPCONN . . . . Y

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC34

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

```

Figure 4-12 CINT: CICS Resources options

In CICS IA 2.2, we can capture detailed information for the following resources:

- ▶ Programs
- ▶ Transactions
- ▶ TDQueues
- ▶ TSQueues
- ▶ Files
- ▶ Webservices
- ▶ Exits

To select detailed information, select option 'D' against the required resource, as shown in Figure 4-12.

To display and modify the DB2/MQ/IMS options, type action code 5 on the line for the DEFAULT CICS region, and press ENTER. See Figure 4-13 on page 83.

```

CIU250          CICS Interdependency Analyzer for z/OS - V2R2M0      2008/04/22
                DB2/MQ/IMS Resource Options for                    01:11:05PM

                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          MQ Options          IMS Options

DB2 . . . . . Y      MQ . . . . . Y      IMS . . . . . Y

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC34
CIU2111I CINT options are updated
F1=          F2=          F3=Exit      F4=          F5=          F6=
F7=          F8=          F9=          F10=         F11=         F12=Cancel

```

Figure 4-13 CINT: DB2/MQ/IMS resource option menu

You can only choose to collect all DB2, WMQ, and IMS resource information. You cannot select to collect DB2 table information only. It is all or nothing for these resource types.

To display and modify the GENERAL options, type action code 6 on the line for the DEFAULT CICS region, and press ENTER. See Figure 4-14 on page 84.

In IA 2.2, two new options were added:

- ▶ Trigger for CINB start
- ▶ Inquire for DB2 resources

We discuss these new options in Chapter 10, “Hints and tips” on page 231.

```

CIU260          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                                     General Options for                  01:12:06PM
          CICS Sysid   : DFTS      CICS Applid   : DEFAULTS

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

Control options
Data to Collect . . . . . : N      (Y=Affinity, N=Interdependency)
Perform periodic saves . . . : Y      (Y=Yes, N=No)
Trigger for CINB start . . . : 5      (2 to 9999 thousand record updates)
Inquire for DB2 resources. . : Y      (Y=Yes, N=No)
Restore data on start . . . : Y      (Y=Yes, N=No)
Multiple signon with same id : N      (Y=Yes, N=No)
Maintain usage counts . . . : N      (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)

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC34

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

```

Figure 4-14 CINT: General Options menu

To display and modify the Affinities options, type action code 7 on the line for the DEFAULT CICS region, and press ENTER. See Figure 4-15 on page 85.

```

CIU270  CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
          CICS Affinities Options for                          01:12:43PM
          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

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

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC34

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

```

Figure 4-15 CINT: CICS Affinities Options menu

To display and modify the time and date options, type action code 8 on the line for the DEFAULT CICS region, and press ENTER. See Figure 4-16 on page 86.

```

CIU280          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                  Time and Date Options for                          01:13:34PM
                  CICS Sysid: DFTS      CICS Applid: DEFAULTS

Modify the options and press Enter to update or F12 to Cancel.

Time and date 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 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

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: Z328      CICS Applid: IYDZZ328      TermID: TC34

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

```

Figure 4-16 CINT: Time and date options

We discuss the time and date options further in Chapter 10, “Hints and tips” on page 231.

If you select option 1 Operations Menu from the Main Administration Menu, Figure 4-17 on page 87 is displayed.


```

CIU100          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                  Operations Menu                                     02:15:25PM

Type action code then press ENTER.                                     More :

1= Start 2= Stop 3= Pause 4= Continue 5= Statistics

      CICS      CICS      Start      Start
Act  Applid    Sysid Status   Date       Time       Collecting
ALL      ALL
IYDZZ138  Z138 UNCONNECTED
IYDZZ228  Z228 UNCONNECTED
IYDZZ238  Z238 UNCONNECTED
IYDZZ318  Z318 UNCONNECTED
IYDZZ328  Z328 STOPPED

CICS Sysid:  Z328   CICS Applid:  IYDZZ328   TermID:  TC34

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

```

Figure 4-17 CINT: Operations menu

Use the operations menu to start, stop, pause, or continue the Collector.

You can perform these four actions against all of the CICS regions by entering the action against the ALL command line.

In Figure 4-17, locate the Act field, which is located on the left of the Applid field, enter action code 5 Statistics, and press ENTER. Figure 4-18 on page 88 is displayed.

```

CIU150          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/22
                Statistics Menu for                                     02:38:35PM

                CICS Sysid   : Z328      CICS Applid   : IYDZZ328

CINT state . . . . . : STOPPED      Collecting Dependencies
Number of pauses . . . . . : 0
Number of saves. . . . . : 1
Records written last save. : 135
Total records on file. . . : 165

Date/time of last start. . : 2008/04/22 08:55:46AM
Date/time of last save . . : 2008/04/22 09:00:08AM
Date/time of last change . : 2008/04/22 08:59:55AM

Total time RUNNING . . . . : 0000:04:19 (HHHH:MM:SS)
Total time PAUSED. . . . . : (HHHH:MM:SS)

Table dataspace name . . . : % full

CICS Sysid: Z328  CICS Applid: IYDZZ328  TermID: TC34

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

```

Figure 4-18 CINT: Statistics menu

4.3 What happens after the Collector is started

After collection starts, the Dependency or Affinity data that is captured is kept in an z/OS data space. At periodic intervals (set by you) or when collection is stopped, the data space is emptied into the CICS IA VSAM files for the region.

Currently the VSAM file holds a First Recorded time and a Change Recorded Time for each resource call in the program. It also holds a Use Count.

The VSAM key contains the offset (location) of the resource call within a program. So if we had two EXEC CICS WRITEQ TS QUEUE('FRED') commands within a program, there would be two records in the VSAM file.

At a time of your choice, the DB2 Load program can process the VSAM file that is provided with CICS IA.

Note: Collection must be stopped when the CICS IA DB2 load program is run.

4.4 The DB2 load programs

The VSAM load jobs run a number of programs to load various DB2 tables. We provide five sample DB2 update jobs for Dependency data:

- ▶ CIUUPDB - loads CICS,WMQ,IMS and DB2 resource information
- ▶ CIUUPDB1- loads CICS resource information
- ▶ CIUUPDB2 - loads DB2 resource information
- ▶ CIUUPDB3 - loads WMQ resource information
- ▶ CIUUPDB4 - loads IMS resource information

These jobs read the relevant VSAM files and update the corresponding DB2 tables.

All of these jobs also update two other CICS IA DB2 tables:

- ▶ CIU_REGION_INFO

This table contains information about the CICS region - CSD name, CSD list names, the time that the dependency was last collected, the time that the Affinity was last collected, and the CICS release level.

- ▶ CIU_RESOURCE

This table was created to improve the performance of the IA Explorer. It combines information from the four main tables:

- CIU_CICS_DATA
- CIU_DB2_DATA
- CIU_MQ_DATA
- CIU_IMS_DATA

The CICS IA Affinity data is loaded by the sample job CIUAFFLD. This job also updates the CIU_REGION_INFO and CIU_RESOURCE tables.

Note: The sample job CIUAFFLX can be used to load Affinity data that is gathered by the CAU function in CICS TS 1.3 and CICS TS 2.2 into the DB2 Affinity tables.



Query interface

In this chapter, we address the partial list of the questions that are referenced in 1.1, “What can CICS IA do for you” on page 2 and show how to do the queries in CICS IA to get the answers. Transaction CINQ starts the query interface.

Important: The CINQ Query Interface has not been updated since IA V1.3 and will not be in the future. All development efforts for new resources are covered by the new IA Explorer and sample SQL queries.

Important: Although we answer the questions that are asked here using the CICS IA online starter-set of queries, there are other more sophisticated queries that you can make using SQL query tools, such as SPUFI or other ISV software that might be in use in your environment. We show some sample SQL queries later in this book.

5.1 The main window after CINQ is entered

Figure 5-1 shows the main window after you enter CINQ.

```
CIU400          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/05/13
                                Query Menu
08:55:40AM

Select one of the following. Then press Enter.

-  1  Inquire on CICS Resources.
    2  Inquire on DB2 Resources.
    3  Inquire on MQ Resources.
    4  Inquire on IMS Resources.
    5  Additional inquire on DB2 Resources.
    6  Inquire on CICS Affinities.
```

Figure 5-1 Main window when CINQ is entered

5.2 What transactions is a given program invoked by

To answer this question:

On the Selection window, having already entered option 1 from the main CINQ window, type option 1 Transactions, as shown in Figure 5-2 on page 93.

```
CIU500          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                                     CICS Query Menu
12:24:13PM

Select the resource type to query:   1

1. Transactions
2. Programs
3. TSQs
4. TDQs
5. Maps
6. Files
7. Applications
8. Regions

OR display all resources in application      detailed N (Y/N)

Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option 7 may take a long time.

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62
```

Figure 5-2 CICS resource: enter option 1 to query on Transactions

Figure 5-3 on page 94 appears and the program named DSWSUBLX is entered.

```
CIU510          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                CICS Query  TRANSACTIONS
12:27:43PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to specify a generic name. Note: Only one option can be chosen.

1. Start transaction....          (eg. TRN1 or TRN%)
2. Used by tran.....
3. Use program.....  DSWSUBLX
4. Used by program.....
5. Use TDQ.....                Detailed TDQ output?  N
6. Use TSQ.....                Detailed TSQ output?  N
7. Use map.....
8. Use file (ddname)....        Detailed file output? N
9. In region.....
10. DTP to transaction...
11. In application.....        (enter ? for a list of application codes)

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-3 CICS resources: selecting to query on program DSWSUBLX

Figure 5-4 on page 95 shows the transactions that use program DSWSUBLX.


```

CIU560          CICS Interdependency Analyzer for z/OS - V2R2M0
For your CICS Query
WHICH TRANS    USE PROGRAM DSWSUBLX

    HOME  PROGRAM  USED BY
    SYSID                TRAN
    PAA1   DSWSUBLX  HX1
                                HX2
                                IX2
                                IX8
                                PX2
                                PX3
                                SX2
                                SX4
                                SX6
    PAA4   DSWSUBLX  HX1
                                HX2
                                IX2
                                IX8

```

Figure 5-4 CICS resources: results for transactions using program DSWSUBLX

5.3 What transactions access a particular file and how

To answer this question:

From the main CICS resources menu, enter option 1 to query regarding Transactions, as shown in Figure 5-5 on page 96.

```
CIU500          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                                CICS Query Menu
12:29:37PM

Select the resource type to query:  1

1. Transactions
2. Programs
3. TSQs
4. TDQs
5. Maps
6. Files
7. Applications
8. Regions

OR display all resources in application      detailed N (Y/N)

Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option 7 may take a long time.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-5 CICS resources: enter option 1 to query on Transactions

In the resulting window, Figure 5-6 on page 97, the file of interest is specified. Also, a Y is entered to indicate that details (type of file access) are desired.

```

CIU510          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                CICS Query  TRANSACTIONS
12:30:41PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to specify a generic name. Note: Only one option can be chosen.

1. Start transaction....          (eg. TRN1 or TRN%)
2. Used by tran.....
3. Use program.....
4. Used by program.....
5. Use TDQ.....                   Detailed TDQ output?  N
6. Use TSQ.....                   Detailed TSQ output?  N
7. Use map.....
8. Use file (ddname).... HOTEL1   Detailed file output? N
9. In region.....
10. DTP to transaction...
11. In application.....          (enter ? for a list of application codes)

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62

```

Figure 5-6 CICS resources: enter file HOTEL1 for the query

The results of the query in Figure 5-5 on page 96 are shown in Figure 5-7.

```

CIU560          CICS Interdependency Analyzer for z/OS - V2R2M0
For your CICS Query
WHICH TRANS    USE FILE HOTEL1

      HOME  FILE    USED BY  FUNCTION
      SYSID
PAA1  HOTEL1  HR1     READ UPD
                        REWRITE
                        HR2     READ
                        WRITE
PAA4  HOTEL1  HR1     READ UPD
                        REWRITE
                        HR2     READ
                        WRITE

```

Figure 5-7 CICS resources: results of query on transactions using file HOTEL1

5.4 What are the resources that a specific program uses

There is no single online query provided to answer this question. You can, however, find out for each type of resource which, if any, are used by the specific program, for example, the following sequence of windows show the map or maps that are used by a program.

Figure 5-8 shows the query on maps.

```
CIU500          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                CICS Query Menu
12:31:26PM

    Select the resource type to query:    5

    1. Transactions
    2. Programs
    3. TSQs
    4. TDQs
    5. Maps
    6. Files
    7. Applications
    8. Regions

    OR display all resources in application      detailed N (Y/N)

    Enter the application's 3 character code or
    ? for a list of applications available.
    WARNING : Option 7 may take a long time.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-8 Query on maps

In Figure 5-9 on page 99, the program of interest is specified.

```
CIU520          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                CICS Query  MAPS
12:32:12PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to select a generic name. Note: Only one option can be chosen.

1. Start transaction....
2. Used by tran.....
3. Use program.....
4. Used by program..... DSWFORVV
5. Use TDQ.....
6. Use TSQ.....
7. Use map.....
8. Use file (ddname)....
9. In region.....
10. DTP to transaction...
11. In application..... (enter ? for a list of application codes)

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-9 Query for maps used by program DSWFORVV

Figure 5-10 on page 100 shows the maps that the program DSWFORVV uses.

```
CIU560          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
  For your CICS Query
12:32:58PM
WHICH MAPS      ARE USED BY PROGRAM DSWFORVV                      Page 1 of 4

    HOME PROGRAM      USES
    SYSID              MAP
    PAA1 DSWFORVV     MAPHR1I
                              MAPHX1I
                              MAPIT1I
                              MAPIT2I
                              MAPIT8I
                              MAPIX1I
                              MAPIX2I
                              MAPIX8I
                              MAPPS2I
                              MAPPS3I
                              MAPPX2I
                              MAPPX3I
MAPSC2I
```

Figure 5-10 Maps used by program DSWFORVV

To continue the process of determining all of the resources that are used by this program, we go back to the main menu, and enter the next resource of interest, in this case option 6, Files, as shown in Figure 5-11 on page 101.

```
CIU500          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                                     CICS Query Menu
12:34:18PM

Select the resource type to query:    6

1. Transactions
2. Programs
3. TSQs
4. TDQs
5. Maps
6. Files
7. Applications
8. Regions

OR display all resources in application    detailed N (Y/N)

Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option 7 may take a long time.

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62
```

Figure 5-11 Query for files used by program DSWFORVV

In the resulting window, Figure 5-12 on page 102, the program is entered.

```

CIU520          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                CICS Query  FILES
12:34:56PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to select a generic name. Note: Only one option can be chosen.

1. Start transaction....
2. Used by tran.....
3. Use program.....
4. Used by program..... DSWFORVV
5. Use TDQ.....
6. Use TSQ.....
7. Use map.....
8. Use file (ddname)....
9. In region.....
10. DTP to transaction...
11. In application.....      (enter ? for a list of application codes)

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62

```

Figure 5-12 Query on files for program DSWFORVV

The results of this query are the files used by program DSWFORVV, as shown in Figure 5-13.

```

CIU560          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
For your CICS Query
12:36:28PM
WHICH FILES ARE USED BY PROGRAM DSWFORVV .....Page 1 of 1

  HOME   PROGRAM   USES
  SYSID          FILE
  PAA1   DSWFORVV HOTEL1
  PAA4   DSWFORVV HOTEL1

```

Figure 5-13 Files used by program DSWFORVV

You can repeat this process for all of the other resources that we might be interested in related to program DSWFORVV.

5.5 How is a file accessed by a particular program

While doing maintenance on a CICS application with which you are not familiar, you might want to understand quickly how a given file is accessed.

1. On the CICS Query menu (Figure 5-14), enter option 2 (Programs).

```
CIU500          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                CICS Query Menu
12:36:29PM

    Select the resource type to query:  2

    1. Transactions
    2. Programs
    3. TSQs
    4. TDQs
    5. Maps
    6. Files
    7. Applications
    8. Regions

    OR display all resources in application      detailed N (Y/N)

    Enter the application's 3 character code or
    ? for a list of applications available.
    WARNING : Option 7 may take a long time.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-14 CICS resources: enter option 2 to query on Programs

2. Type in the name of the of the file, in this case HOTEL1, as shown in Figure 5-15. Select **Y** for details to show the access types.

```
CIU510          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                  CICS Query  PROGRAMS
12:36:58PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to specify a generic name. Note: Only one option can be chosen.

1. Start transaction....          (eg. TRN1 or TRN%)
2. Used by tran.....
3. Use program.....
4. Used by program.....
5. Use TDQ.....                   Detailed TDQ output?  N
6. Use TSQ.....                   Detailed TSQ output?  N
7. Use map.....
8. Use file (ddname)....  HOTEL1   Detailed file output? N
9. In region.....
10. DTP to transaction...
11. In application.....          (enter ? for a list of application codes)

CICS Sysid:  PAA1  CICS Applid:  SCSCPAA1  TermID:  TC62
```

Figure 5-15 CICS resources: enter the file to query

Figure 5-16 on page 105 shows the results.

HOME	PROGRAM	USES	FUNCTION
		FILE	
PAA1	DSWHR1VV	HOTEL1	READ UPD REWRITE
	DSWHR2VV	HOTEL1	READ WRITE
PAA4	DSWHR1VV	HOTEL1	READ UPD REWRITE
	DSWHR2VV	HOTEL1	READ WRITE

Figure 5-16 CICS resources: results of query on file access by programs

5.6 Which affinities does a transaction have

In this example, we look at transactions and search for a particular type of affinity: Getmain/Freemain. Then we look at what transaction and system affinities exist.

1. From the main window, Figure 5-17 on page 106, type Affinities.
2. Enter option 1, Transactions, as shown in Figure 5-18 on page 107.

CIU400 CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

Query Menu

12:41:14PM

Select one of the following. Then press Enter.

- 6 1 Inquire on CICS Resources.
- 2 Inquire on DB2 Resources.
- 3 Inquire on MQ Resources.
- 4 Inquire on IMS Resources.
- 5 Additional inquire on DB2 Resources.
- 6 Inquire on CICS Affinities.

CICS Sysid: PAA1 CICS Applid: SCSCPAA1 TermID: TC62

Figure 5-17 Affinities

CIUA00 CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

CICS Affinities Query Menu

12:43:35PM

Select the Affinty type to query: 1

- | | |
|----------------------------------|------------------------|
| 1. Transactions | 9. Load Hold/Release |
| 2. Programs | A. Retrieve/Wait/Start |
| 3. ENQ/DEQ | B. Cancel |
| 4. TSQueues | C. CWA |
| 5. Getmain/Freemain | D. Transaction System |
| 6. Getmain/Freemain(Unmatched) | E. Applications |
| 7. Load Hold/Freemain | F. Regions |
| 8. Load Hold/Freemain(Unamtched) | G. Groups |

OR display all affinities in application
Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option C may take a long time.

CICS Sysid: PAA1 CICS Applid: SCSCPAA1 TermID: TC62

Figure 5-18 Affinities: enter option 1 to query on Transactions

3. Enter the type of affinity. In this case we enter option 3, Getmain/Freemain, as shown in Figure 5-19.

```
CIUA10      CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                Affinity Query  TRANSACTIONS
12:44:05PM

Select the Affinty type to query:    3

    1. Has Affinity ENQ/DEQ
    2. Has Affinity TSQueues
    3. Has Affinity Getmain/Freemain
    4. Has Affinity Getmain/Freemain(Unmatched)
    5. Has Affinity Load Hold/Freemain
    6. Has Affinity Load Hold/Freemain(Unmatched)
    7. Has Affinity Load Hold/Release
    8. Has Affinity Retrieve/Wait/Start
    9. Has Affinity CANCEL
    A. Has Affinity CWA
    B. Has Transaction System Affinites

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-19 Affinities: enter option 3 to query on Getmain/Freemain affinities

The results, Figure 5-20, show that two transactions, COIO and CONL, have a Getmain or a Freeman Affinity.

```
CIUA60      CICS Interdependency Analyzer for z/OS - V2R2M0
2008/07/27
For your Affinities Query
12:44:48PM
WHICH TRANSACTIONS HAVE GETMAIN AFFINITIES
1
Page 1 of 1

HOME      AFFINITY  TRAN  COMMAND
APPLID    GROUP
SCSCPAA1  GM.0000002  COIO  GETMAIN
          GM.0000002  CONL  FREEMAIN
SCSCPAA4  GM.0000001  COIO  GETMAIN
          GM.0000001  CONL  FREEMAIN
```

Figure 5-20 Affinities: results of query on Getmain/Freemain affinities

4. To find transaction/system affinities, we go back to the main affinities window, and enter Transactions (option 1) again, as shown in Figure 5-21.

```
CIUA00          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                CICS Affinities Query Menu

12:46:42PM

Select the Affinty type to query:   1

1. Transactions                    9. Load Hold/Release
2. Programs                       A. Retrieve/Wait/Start
3. ENQ/DEQ                        B. Cancel
4. TSQueues                       C. CWA
5. Getmain/Freemain              D. Transaction System
6. Getmain/Freemain(Unmatched)   E. Applications
7. Load Hold/Freemain            F. Regions
8. Load Hold/Freemain(Unamtched) G. Groups

OR display all affinities in application
Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option C may take a long time.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-21 Affinities: enter option 1 to query on transactions

5. Enter option 8 to query on transaction/system affinities, as shown in Figure 5-22.

```
CIUA10          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27
                Affinity Query  TRANSACTIONS
12:47:22PM

Select the Affinty type to query:    8

    1. Has Affinity ENQ/DEQ
    2. Has Affinity TSQueues
    3. Has Affinity Getmain/Freemain
    4. Has Affinity Getmain/Freemain(Unmatched)
    5. Has Affinity Load Hold/Freemain
    6. Has Affinity Load Hold/Freemain(Unmatched)
    7. Has Affinity Load Hold/Release
    8. Has Affinity Retrieve/Wait/Start
    9. Has Affinity CANCEL
    A. Has Affinity CWA
    B. Has Transaction System Affinites

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-22 Affinities: enter option 8 to query on transaction/system affinities

Figure 5-23 shows the results.

```
CIUA60          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/07/27
  For your Affinities Query
12:47:58PM
WHICH TRANSACTIONS HAVE TRANSACTION-SYSTEM AFFINITIES           Page 1 of 6
```

HOME	TRAN	AFFINITY	COMMAND	AFFINITY
APPLID		GROUP		TYPE
SCSCPAA1	CINB	EX.0000002	EXTRACT	EXIT
		IN.0000002	INQUIRE	SYSTEM
		WA.0000002	WAIT	EVENT
	CINT	EX.0000002	EXTRACT	EXIT
		IN.0000002	INQUIRE	FILE
				PROGRAM
				SYSTEM
	COIE	CO.0000002	COLLECT	STATS
		IN.0000002	INQUIRE	DSNAME
				EXITPROG
				IRC
				SYSSUMPC
				SYSTEM

Figure 5-23 Affinities: results on query on transaction/system affinities

5.7 How would you query on DB2, WMQ, or IMS

1. From the main query menu, enter which resource to query on. The first one that we address here is DB2 resources, so you enter option 2, as shown in Figure 5-24.

```
CIU400          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                Query Menu

12:49:08PM

Select one of the following.  Then press Enter.

2  1  Inquire on CICS Resources.
   2  Inquire on DB2 Resources.
   3  Inquire on MQ Resources.
   4  Inquire on IMS Resources.
   5  Additional inquire on DB2 Resources.
   6  Inquire on CICS Affinities.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-24 DB2 resources: enter option 2 to query DB2 Resources

2. To see all of the transactions that access DB2 resources, type the wild card character (%) in option 1, Used by tran field, as shown in Figure 5-25.

```
CIU620          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                DB2 Query Menu

12:49:45PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to select a generic name. Note: Only one option can be chosen.

  1. Used by tran..... %
  2. Used by program.....
  3. Used by plan.....
  4. In region.....
  5. In DB2ID.....

OR display all resources in application

Enter the application's 3 character code or
? for a list of applications available.
WARNING : Option 4 and 5 may take a long time.

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62
```

Figure 5-25 DB2 resources: selecting to query on all transactions accessing DB2

Figure 5-26 on page 114 shows the results.

HOME	TRAN	PROGRAM	DB2ID	PLAN	SECTION	STATEMENT
SYSID					NUMBER	NUMBER
PJA6	DB2N	PROGDB2N	D7Q2	PROGDB2N	00001	00098
			D7Q2	PROGDB2N	00001	00104
			D7Q2	PROGDB2N	00001	00132
	DB2R	PROGDB2R	D7Q2	PROGDB2R	00001	00097
			D7Q2	PROGDB2R	00001	00103
			D7Q2	PROGDB2R	00001	00136
	DB2U	PROGDB2U	D7Q2	PROGDB2U	00001	00319
			D7Q2	PROGDB2U	00002	00387
PJA7	DB2N	PROGDB2N	D7Q2	PROGDB2N	00001	00098
			D7Q2	PROGDB2N	00001	00104
			D7Q2	PROGDB2N	00001	00132
	DB2R	PROGDB2R	D7Q2	PROGDB2R	00001	00097
			D7Q2	PROGDB2R	00001	00103

Figure 5-26 DB2 resources: results from query on transactions accessing DB2

3. To obtain what transactions use MQ queues, on the Query menu, enter option 3 for Inquire on MQ Resources, as shown in Figure 5-27.

```
CIU400          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                Query Menu
12:51:13PM

Select one of the following.  Then press Enter.

3  1  Inquire on CICS Resources.
   2  Inquire on DB2 Resources.
   3  Inquire on MQ Resources.
   4  Inquire on IMS Resources.
   5  Additional inquire on DB2 Resources.
   6  Inquire on CICS Affinities.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-27 MQ resources: enter option 3, Inquire on MQ Resources

4. Enter option 1 for querying on Transactions, as shown in Figure 5-28.

```
CIU700          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                MQ Query Menu

12:51:46PM

    Select the resource type to query:    1

    1. Transactions
    2. Programs
    3. Queues
    4. Applications
    5. Regions

    OR display all resources in application

    Enter the application's 3 character code or
    ? for a list of applications available.
    WARNING : Option 5 may take a long time.

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC62
```

Figure 5-28 MQ resources: enter option 1 to query on Transactions

5. To see all of the transactions that access queues, type the wildcard character (%), as shown in Figure 5-29.

```
CIU710          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                TRANSACTIONS

12:52:34PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to specify a generic name.

1. Use Queue ..... %
```

Figure 5-29 MQ resources: using the wildcard character to see all transactions

Figure 5-30 on page 117 shows the results.

CIU760		CICS Interdependency Analyzer for z/OS - V2R2M0	
2008/04/27			
For your MQ Query			
12:53:22PM			
WHICH TRANS		USE MQ QUEUE %	Page 1 of 1
HOME	TRAN	USES	
SYSID		QUEUE	
TSTD	MAIL	CSQ4SAMP.BBBAFEA9EDFBEE85	
		CSQ4SAMP.MAILMGR.DJWHITE	
		CSQ4SAMP.MAILMGR.DJWHITE.EMJ	
		CSQ4SAMP.MAILMGR.DJWHITE.EMYR	
		CSQ4SAMP.MAILMGR.DJWHITE.EM2	
		SYSTEM.COMMAND.INPUT	
	MVGT	DAVES.QUEUE	
	MVPT	DAVES.QUEUE	

Figure 5-30 MQ resources: results of query on transactions accessing queues

- To see all transactions that use IMS, return to the Main Query menu, and enter option 4, Inquire on IMS Resources, as shown in Figure 5-31 on page 118.

CIU400 CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

Query Menu

12:55:19PM

Select one of the following. Then press Enter.

- 4 1 Inquire on CICS Resources.
- 2 Inquire on DB2 Resources.
- 3 Inquire on MQ Resources.
- 4 Inquire on IMS Resources.
- 5 Additional inquire on DB2 Resources.
- 6 Inquire on CICS Affinities.

CICS Sysid: PAA1 CICS Applid: SCSCPAA1 TermID: TC62

Figure 5-31 IMS resources: enter option 4 to query on transactions using IMS

7. Enter option 1 to choose to inquire on Transactions, as shown in Figure 5-32.

```
CIU800          CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                                IMS Query Menu
12:56:08PM

    Select the resource type to query:    1

    1. Transactions
    2. Programs
    3. PSBs
    4. PCBs
    5. Applications
    6. Regions

    OR display all resources in application

    Enter the application's 3 character code or
    ? for a list of applications available.
    WARNING : Option 5 may take a long time.

CICS Sysid: PAA1  CICS Applid: SCSCPAA1  TermID: TC62
```

Figure 5-32 IMS resources: enter option 1 to query on Transactions

8. If you are interested in all PCBs used by transactions, you can use the wildcard character (%) in the Use PCB field, as shown in Figure 5-33 on page 120; otherwise, you could specify the specific PCB name. To get all of the details, specify Y in answer to the question on the same line.

```

CIU810      CICS Interdependency Analyzer for z/OS - V2R2M0
2008/04/27

                IMS Query  TRANSACTIONS

12:57:02PM

Select the option that best suits your query by entering the resource
details in the field provided for each option. Use % as a wildcard
character to specify a generic name.

1. Use PSB .....
2. Use PCB ..... %                Detailed PCB output?  N

```

Figure 5-33 IMS resources: selecting to see all PCBs, with details

Figure 5-34 shows the results.

```

CIU860      CICS Interdependency Analyzer for z/OS
For your IMS Query
WHICH TRANS  USE PCB %

      HOME   TRAN   USES      FUNCTION
      SYSID  SYSID  PCB
      TLS3   ASME   DI21PART  GET NEXT
                        GET UNIQUE
      TLS4   DLE1   MDLHDAM   DELETE
                        GET UNIQUE
                        INSERT
                        REPLACE
                        GET NEXT
                        GET UNIQUE
                        DELETE
                        GET UNIQUE
                        INSERT
                        GET UNIQUE
                        GET UNIQUE
      DLE2   DLE2   MDLHDAM   GET NEXT
                        GET UNIQUE
      DLE3   DLE3   MDLHDAM   DELETE
                        GET UNIQUE
                        INSERT
                        GET UNIQUE
      DLE4   DLE4   MDLHIDAM  GET UNIQUE
      DLE5   DLE5   MDLHIDAM  GET UNIQUE

```

Figure 5-34 IMS resources: results to query on PCB resources



Analyzing CICS IA data using the CICS IA Explorer

The CICS IA Explorer provides a graphical user interface (GUI) to CICS IA that runs toolbar searches on common resources and helps you to build detailed queries to interrogate the Dependency and Affinity database objects. The CICS IA Explorer also provides a number of sample queries.

6.1 CICS IA Explorer overview

Figure 6-1 show all of the views that are available in the new CICS IA Explorer.

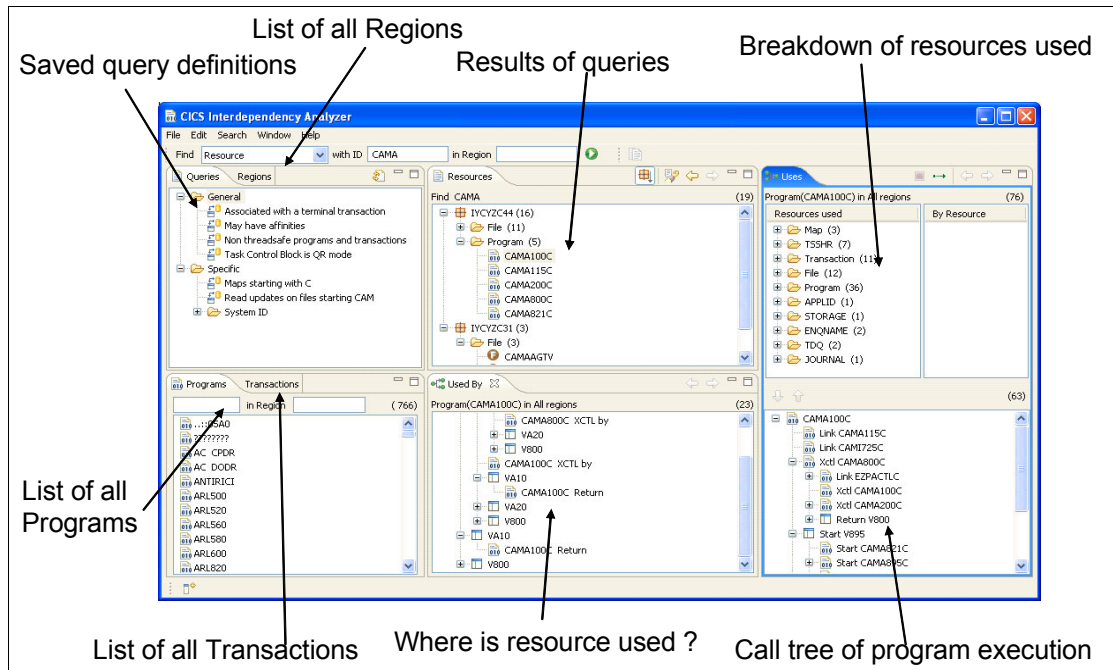


Figure 6-1 Views of CICS IA Explorer

6.2 Toolbar searches and queries

The CICS IA Explorer has two methods of interrogating data that the CICS IA Collector collects: toolbar searches and queries.

6.2.1 Toolbar searches

The CICS IA Explorer has four toolbars that you can use to search for common resources, programs, transactions, and regions, which you can see in Figure 6-2. Toolbar searching is a quick way of interrogating a limited amount of data because the resources that you can interrogate are fixed by the menu options.

You can search by name and region only, or you can use the Regions toolbar to search by regions only. However you can restrict or expand the search criteria by using the wildcard. The structure of the results that are displayed in the Resources window is also fixed. The description of the toolbar search that you run is shown above the results in the Resources window. You cannot save the results of a toolbar search. Figure 6-2 shows the toolbar search.

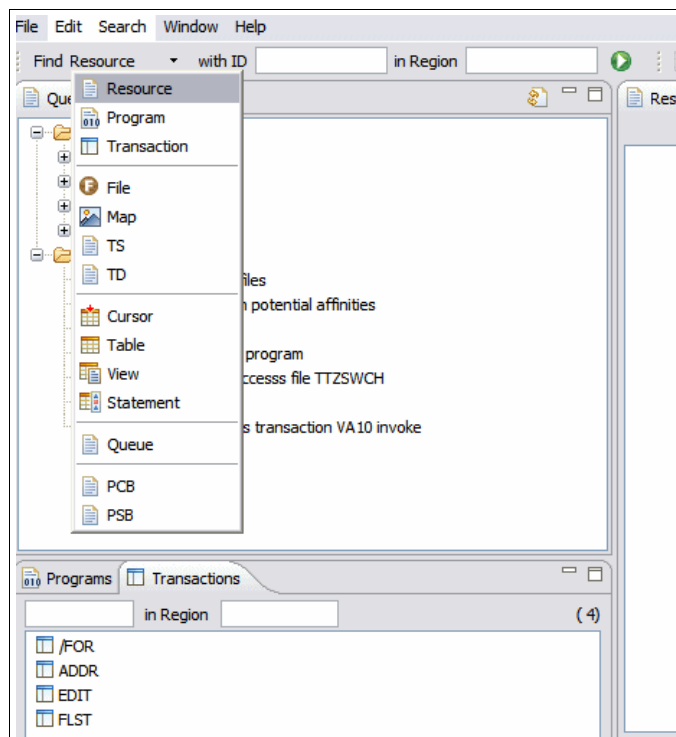


Figure 6-2 Toolbar search

6.2.2 Queries

The CICS IA Explorer provides you with a number of defined queries, which you can edit, and helps you to build detailed queries of your own. When you create a query, you can define the structure of the results that are displayed in the Resources window. The structure is not fixed. You can query any resources in any combination. You can also add filters to the query to further refine the results. The description of the query that you run is shown above the results in the Resources window. You can save queries that you edit or create.

When you run a query, you can save the search results in a list under the query name in the Queries window for future reference. You can copy the definition of a query to the clipboard and paste it into the same folder, or a different folder, in the Query window. If you want to run your own SQL, you can paste the query definition into a text editor to view the raw SQL strings that run against the DB2 tables. Figure 6-3 shows the CICS IA Explorer main window.

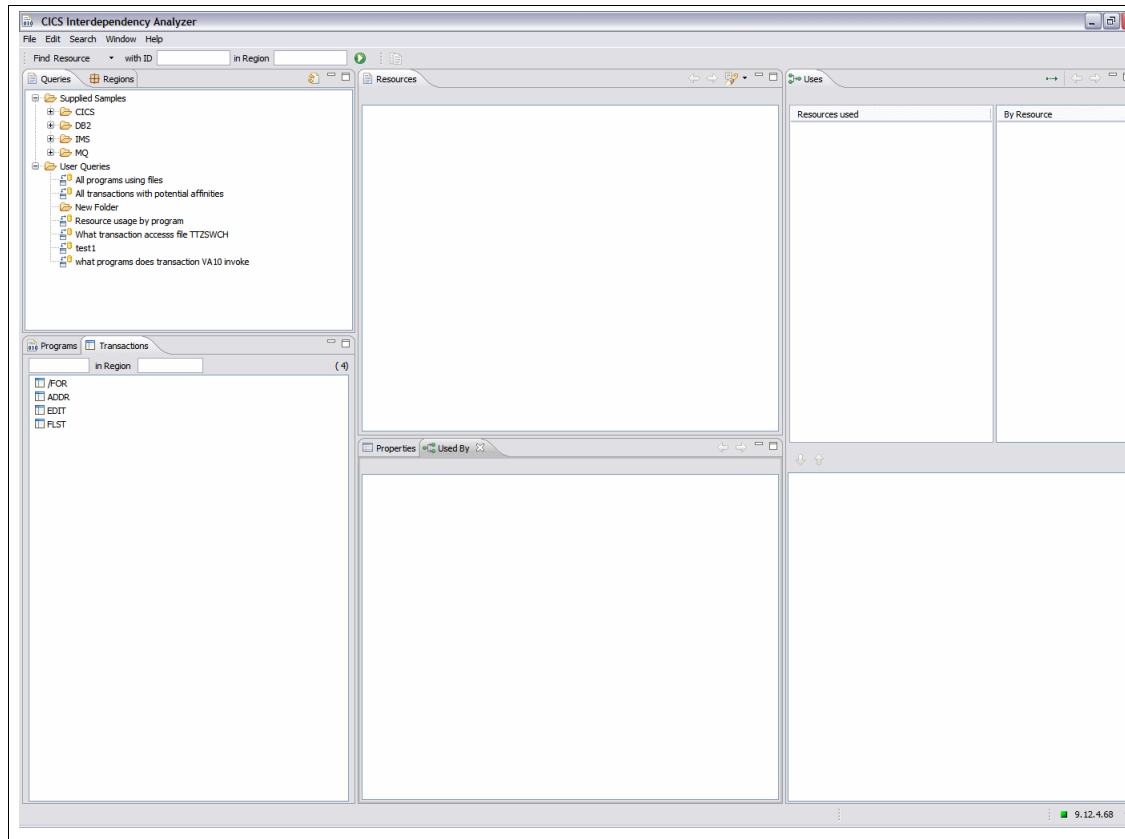


Figure 6-3 Main CICS IA Explorer menu

The Explorer has an implicit wildcard (*), which is added to the end of all search values. The wildcard is not visible, for example, when you search using the toolbar or define queries, if you type FIN as a value, it is read as “FIN*”. The wildcard is denoted by the *. In this way, the search is for all resources that begin with “FIN”.

If you require an explicit wildcard, you can add it anywhere in the search definition. However, the implicit wildcard at the end of the search value remains, for example:

- ▶ To search for resources that begin with “FIN” and include 01, enter FIN*01.
- ▶ To search for resources that contain “01”, enter *01.
- ▶ To narrow your search as much as you want, you can use as many wildcards as required, for example, A*0*3* is also acceptable.

The implicit wildcard is nearly always added to the end of the search value, for example, searching for PROG*AF can return the following list of items:

Note: A wild card is not always added. If it is a transaction, and we enter FINA then we do not add an *.

- ▶ PROGAAAF
- ▶ PROGAafa
- ▶ PROGAFAA

Use the toolbar at the top of the CICS IA Explorer to search for common resources. You can search by name and region only. You can restrict or expand the search using the wildcard.

To run a toolbar search for common resources:

1. Use the menu in the Find field to select a resource to search.
2. Use the with ID field to limit the search to a particular ID, or you can leave the field blank.
3. Use the in Region field to limit the search to a particular region or regions.
4. To run the search, click **Search** → **Run**, or click the green toolbar button, or press **Enter**. The result of the search is shown in the Resources window.

6.3 Programs and Transactions windows

When you first open the CICS IA Explorer and connect to the database the information in the Programs window and Transactions window is retrieved from

the data that is extracted to the database by the CICS IA Collector. The information is displayed in an alphabetic list and is static.

You can use the toolbar search, which is located at the top of the Queries and Regions window, to explore the programs and transactions. Figure 6-4 shows an example of the Programs and Transactions window.

The information in the Programs window is a list of all programs that are known to CICS IA and includes:

- ▶ Programs that are the sources of interactions
- ▶ Programs that are the results of interactions
- ▶ Programs that are CICS resource types

The information in the Transactions window is a list of all of the transactions that are known to IA, and includes:

- ▶ Transactions in which programs are running
- ▶ Transactions that are the results of interactions
- ▶ Transactions that are CICS resources of TRANSID type

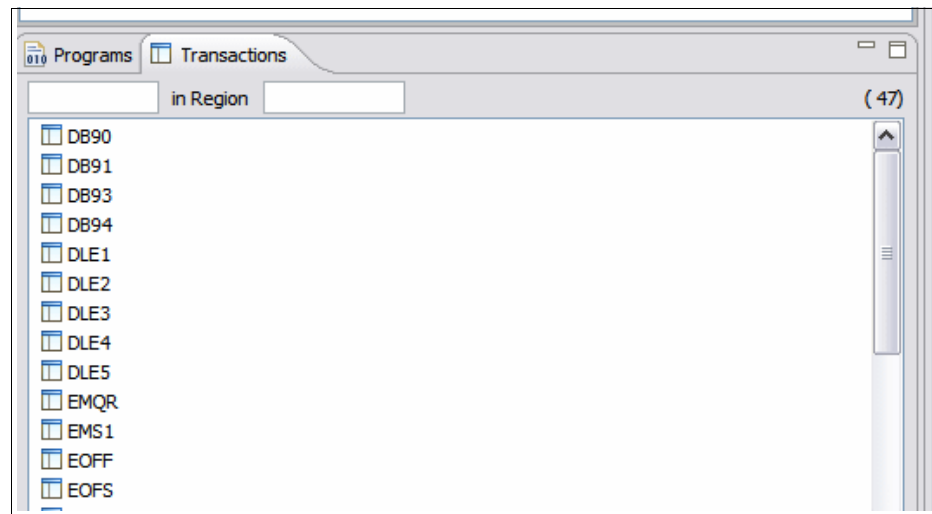


Figure 6-4 Programs and Transactions window

6.3.1 Searching programs and transactions

You can search the programs that are listed in the Programs window. The data is stored in the database by the CICS IA Collector and is retrieved when you first open the CICS IA Explorer and connect to the database.

To search the programs list for particular programs that you want to analyze further:

1. Click the Programs tab to bring the Programs window to the front.
2. To filter the search, in the left field, enter the name of the search criteria, and in the Region field, enter the region search criteria. An implicit wildcard is appended to the end of the search criteria, and you can add wildcards at the beginning and in the middle. You can search by name only, region only, or name and region.
3. Press **Enter**, or click **Search** → **Run** to run the search. The results are displayed in the Programs window.

Note: This is the same process for transactions except that you search transactions in the Transactions window.

6.4 Regions window

When you run a toolbar search or a query, the Regions window displays all of the regions for which the CICS IA Collector stored data in the database. The results are displayed in an alphabetic list. You have only one search field, which is for the region name. Figure 6-5 on page 128 shows an example of this window.

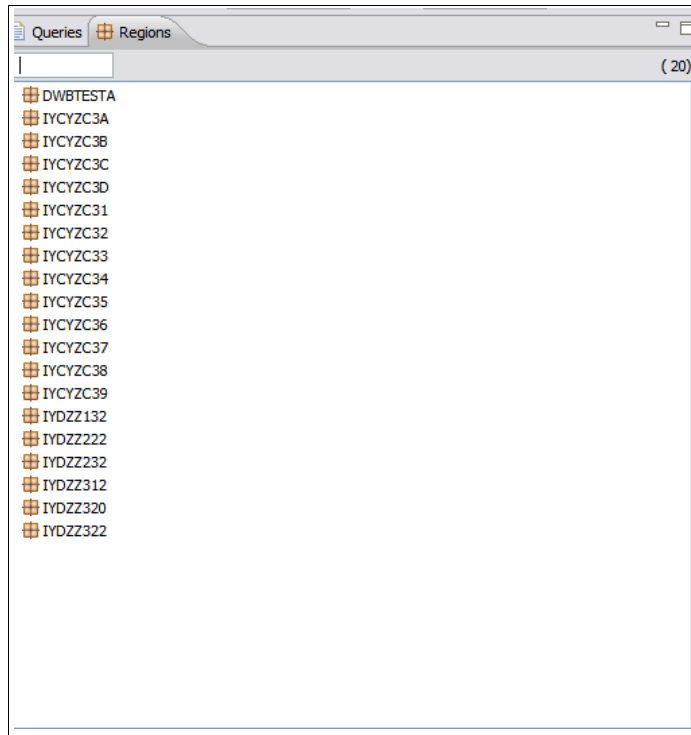


Figure 6-5 Regions window

You can search the regions list for particular transactions that you want to analyze further. To search in the Regions window:

1. Click the Regions tab to bring the Regions window to the front.
2. To filter the search, in the search field, enter the region name search criteria. An implicit wildcard is appended to the end of the search criteria, and you can add wildcards at the beginning and in the middle.
3. Press **Enter** or click **Search** → **Run** to run the search. The results are displayed in the Transactions window.
4. Right-click the name of the region you want to analyze.
5. From the menu, select the resource that you want to search.

6.5 Supplied queries with CICS IA Explorer

There are a number of predefined queries that are supplied with the CICS IA Explorer for each resource. You can edit the queries to suit your requirements.

Figure 6-6 and Figure 6-7 on page 130 show examples of the window you see with the queries all expanded.

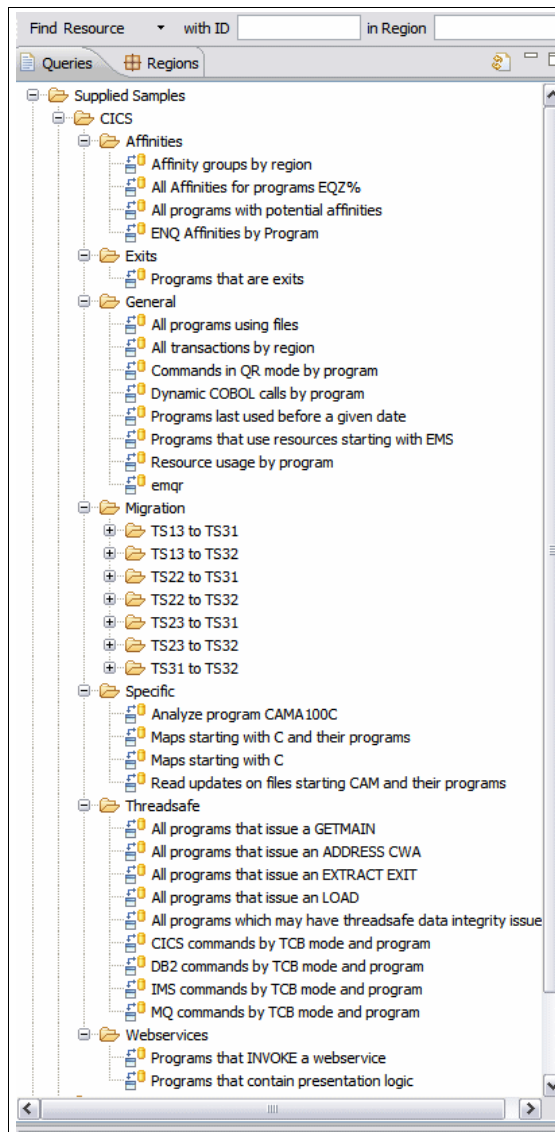


Figure 6-6 CICS supplied queries

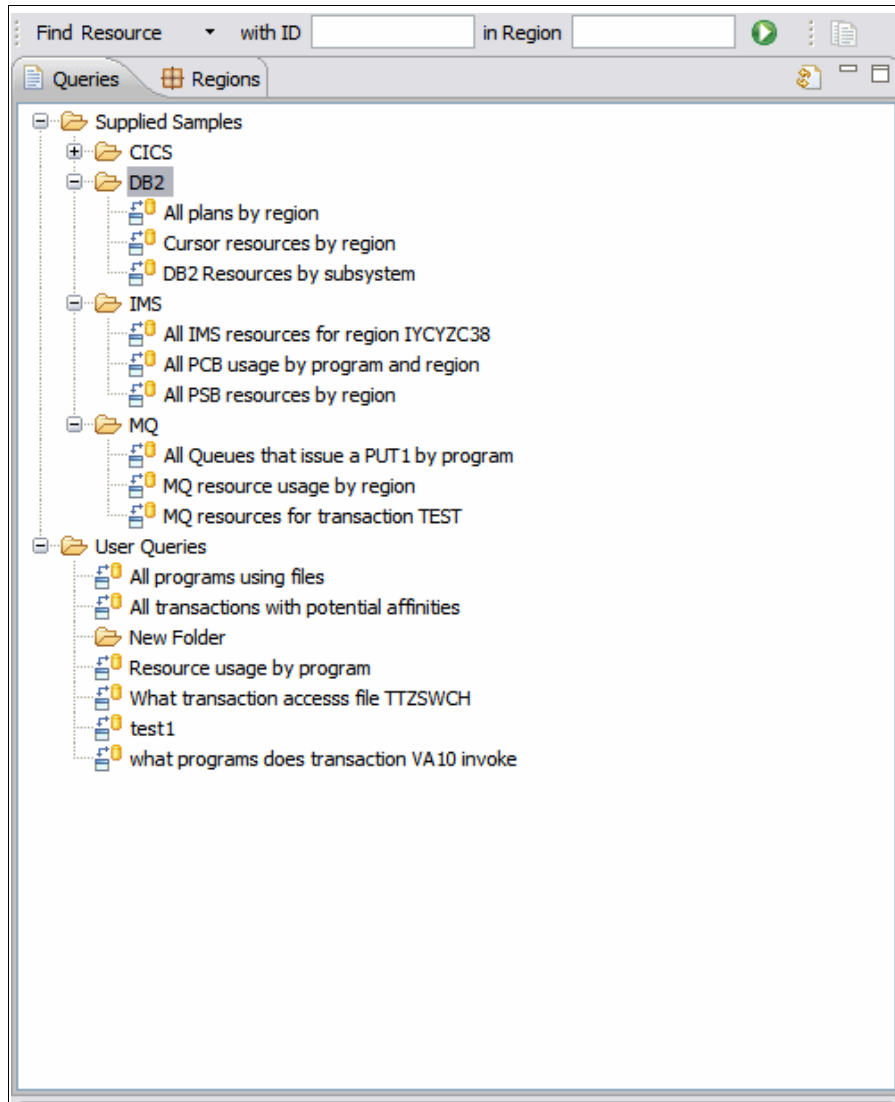


Figure 6-7 DB2 IMS and MQ supplied queries

To run a predefined query in the Queries window:

1. In the Queries window, select the resource that you want to query, as shown in Figure 6-8 on page 131.
2. To view the query names, click the plus sign to expand the list of queries, as shown in Figure 6-9 on page 132.

3. To run the query, right-click the query name, and click **Run**, or click the query you want to run, and click **Search** → **Run**. The results are displayed in the Resources window. You can save the results for future reference and further analyze the results. Figure 6-10 on page 133 is an example of you clicking **Run**, and Figure 6-11 on page 134 shows the results.

Next, we show an step-by-step example of running the supplied query, all transactions by region. Figure 6-8 shows the selected CICS queries.

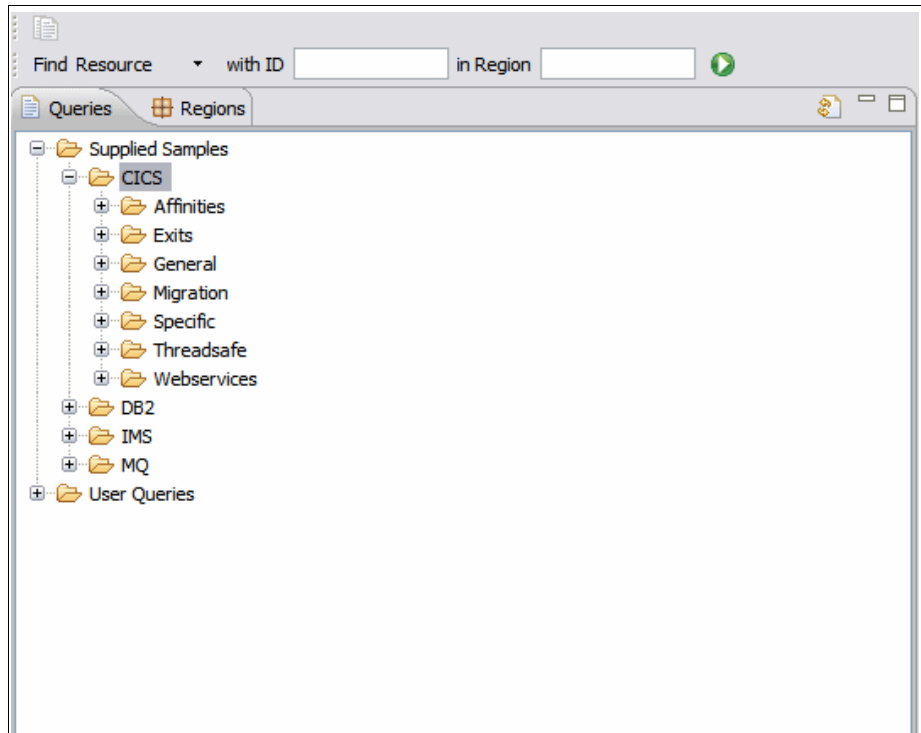


Figure 6-8 CICS query selected

Next, we look at general queries underneath the CICS heading. Figure 6-9 on page 132 shows general queries expanded.

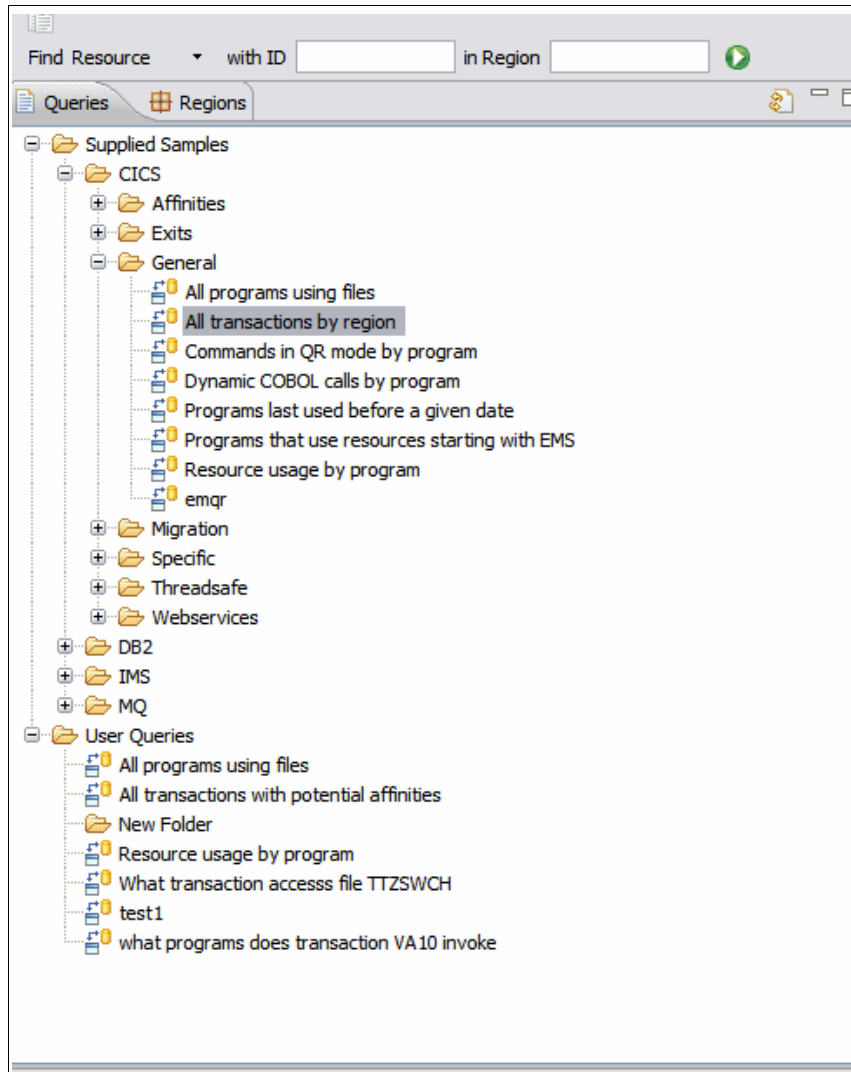


Figure 6-9 General queries expanded

Click **Run** to execute the query, as shown in Figure 6-10 on page 133.

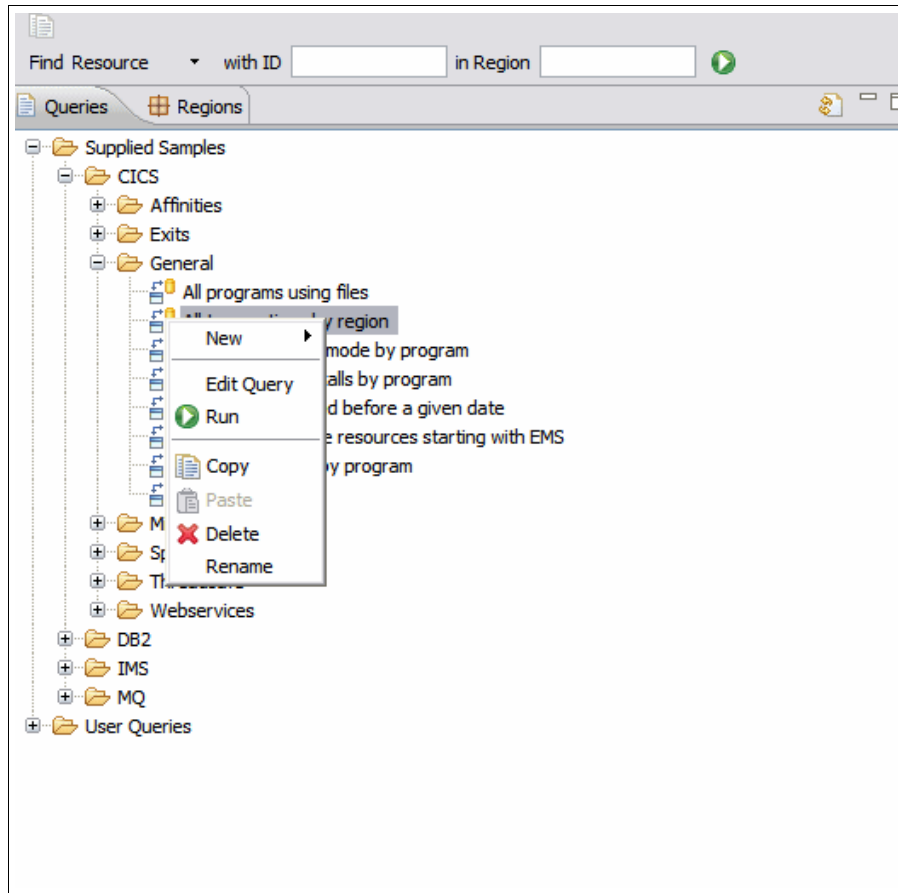


Figure 6-10 click run to execute

Figure 6-11 on page 134 shows results of the all transactions by region query.

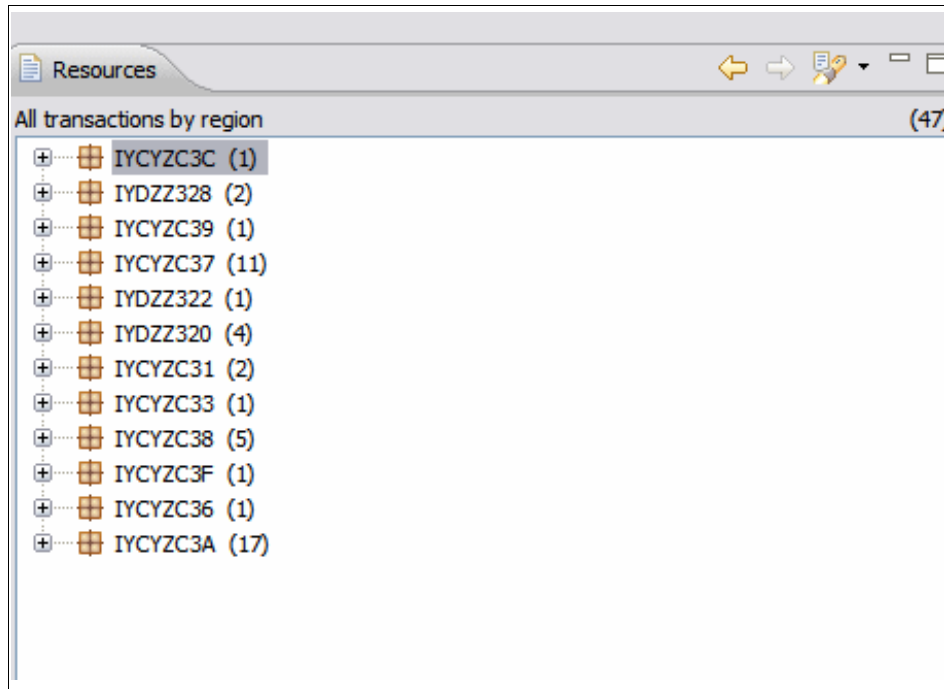


Figure 6-11 all transactions by region results

6.6 Creating your own queries in CICS IA Explorer

To create a query:

1. Click **File** → **New**, or right-click in the Queries window, and select **New**, as shown in Figure 6-12 on page 135. If you want to save the query, right-click the folder name under which you want the query to be listed.

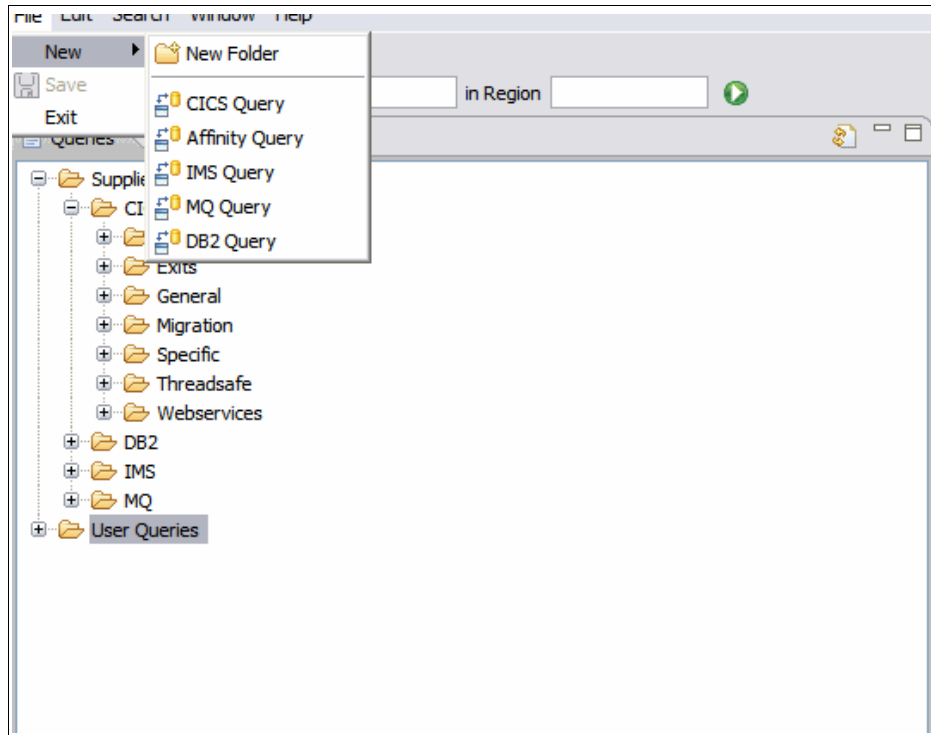


Figure 6-12 Create new query

2. From the menu, select the resource type that you want to query. We selected CICS, as shown in Figure 6-13 on page 136.

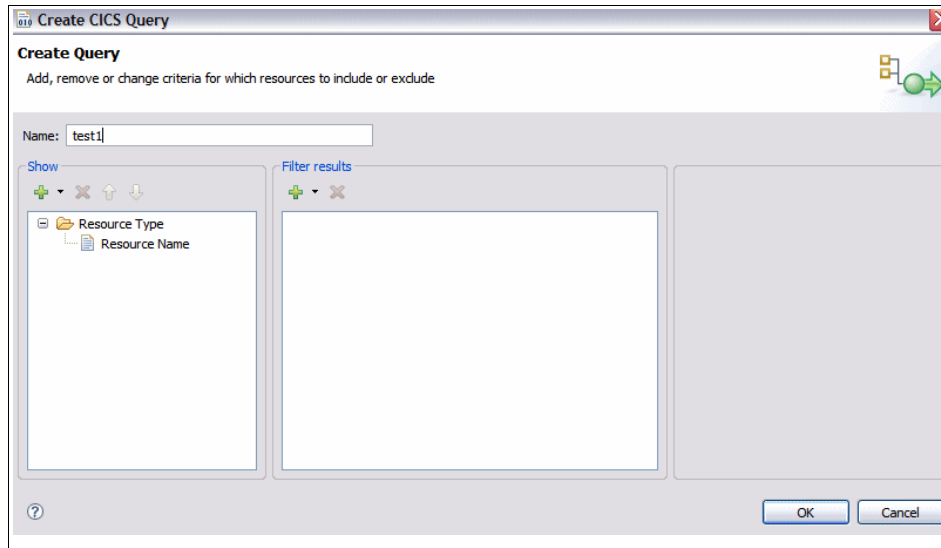


Figure 6-13 Create CICS query

3. In the Create Query window, go to the Name field, and enter a query name. We called ours test1.
4. In the Show panel, click the green plus sign button to select the resource or resources that you want to query, as shown in Figure 6-14 on page 137.

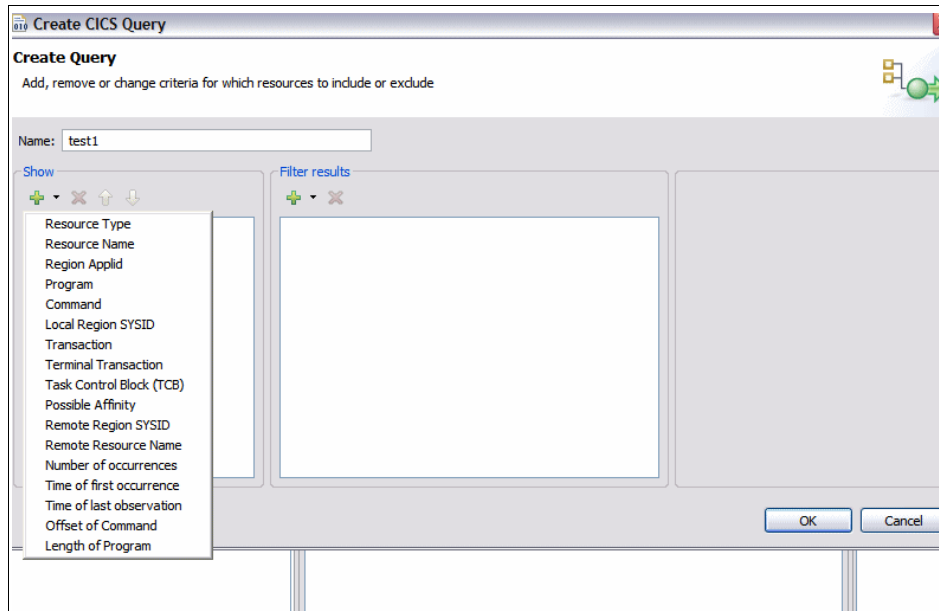


Figure 6-14 Query resource selection

5. Click the red X button to remove resources.
6. Click the yellow arrows to move the selected resources up and down the tree so that the query results are presented in the required order. The resources tree that is created from your selections is in the format in which the results are displayed in the Resources view when you run the query.

6.6.1 Adding filters to the query

When you create a query, you can further refine your query by editing the right pane in the Create Query window. When you select an expression, the right pane changes allowing you to further define your query:

1. Select **is** or **is not** for the values in the query. If the expression you selected has a known set of values, for example, Resource Type, the values are listed. These properties are added to the left pane so that you can view the query string for each line easily. If you need both is and is not values to be included in the query for the same resource, you require two lines for the same resource because you cannot combine is and is not values.
2. Define the expressions you selected by arbitrary values, for example, Resource Name. Use the editable table to define the values. These properties are added to the left pane so that you can easily view the query

string for each line. An implicit wildcard is appended to the end of the value, and you can add wildcards at the beginning and in the middle.

To our test1 query, we selected resources as being transactions. We then further filtered this down to being a particular transaction. So for the filter, we select transaction. We then select **transaction is EMS1**, as shown in Figure 6-15.

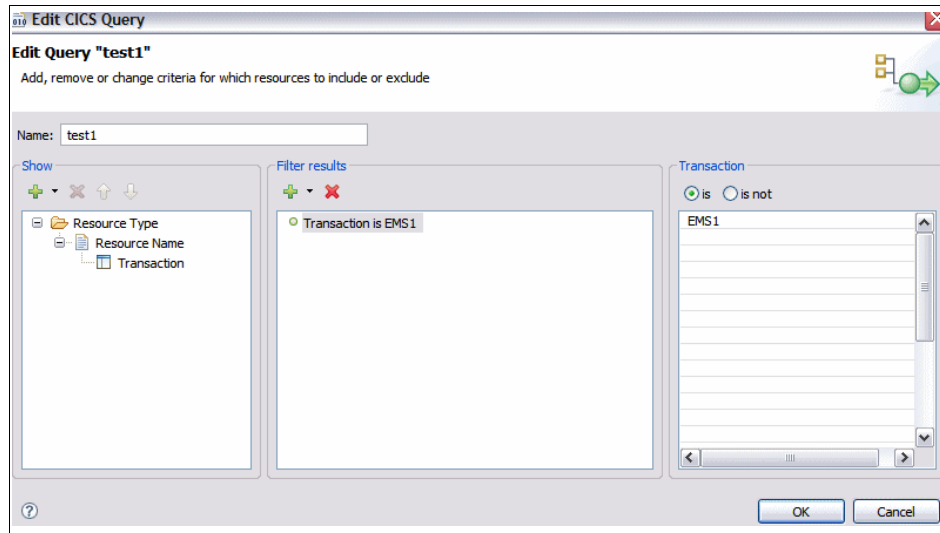


Figure 6-15 Filter query

When we run this query, we get the results shown in Figure 6-16 on page 139.

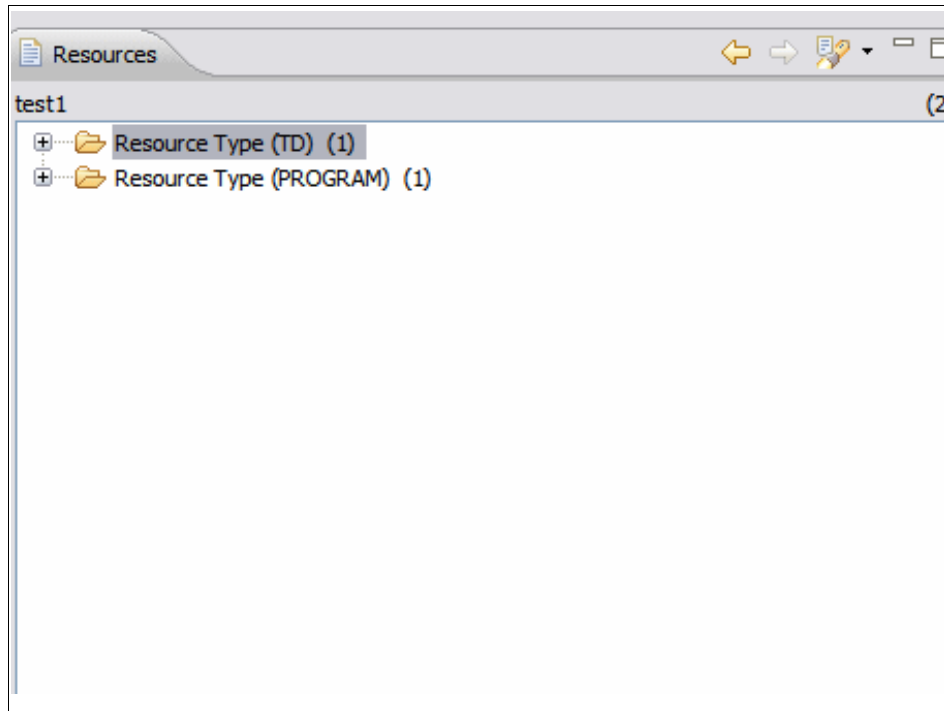


Figure 6-16 test1 results

6.6.2 Modifying an existing query

You can modify existing queries, whether they are supplied predefined queries or queries that you created.

To modify an existing query:

1. Right-click the query that you want to edit, and click **Edit**, or click the query that you want to edit, and click **Edit** → **Edit Query**. The Edit Query window is displayed, as shown in Figure 6-17 on page 140.

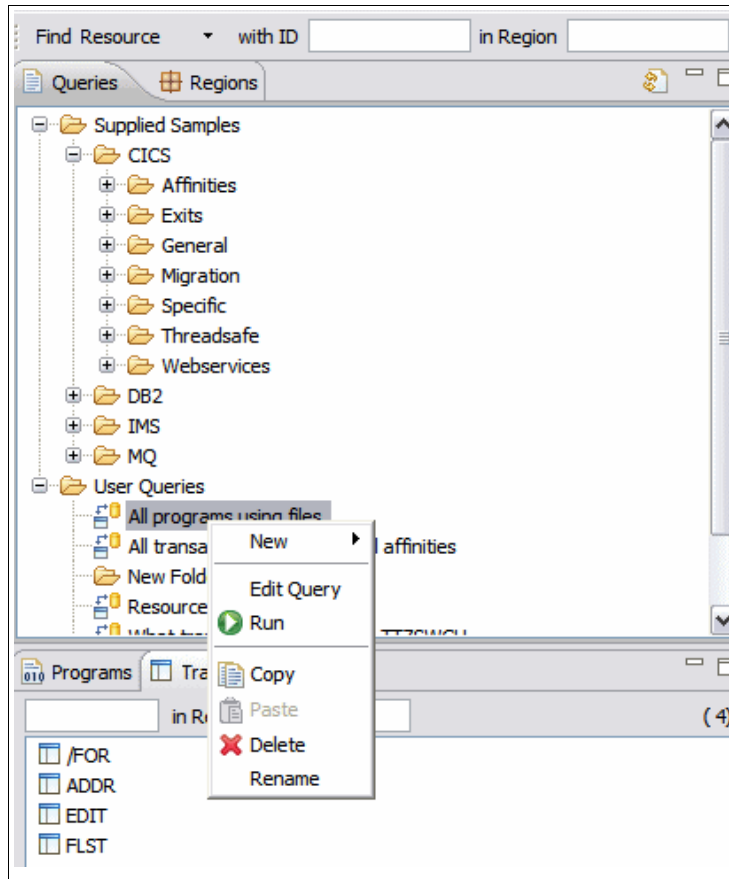


Figure 6-17 Edit query

2. Edit any aspect of the query that you want. To save the edited query as a new query, change the name. Figure 6-18 on page 141 shows the Edit CICS query window.

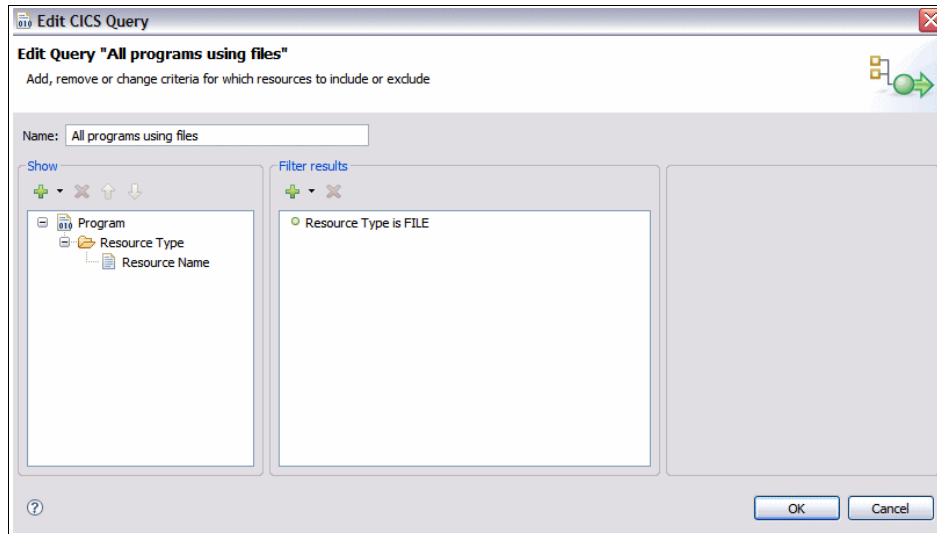


Figure 6-18 Edit CICS query window

3. Click **OK** to save the query. The query is saved in the same folder that the original query is saved.

6.6.3 Saving your queries in folders

When you create and define your queries, you can save them in the Queries window for future use.

To save your queries:

1. Before you create a query, in the Queries window, click the folder name under which you want the query to be listed. You cannot move a query when after it is saved.
2. After you name and define your query, click **OK**, and the query is saved in the query folder that you selected.
3. To delete a query, in the Queries window, right-click the query name, and click **Delete**.

6.6.4 Creating a query folder

You can save queries and query results in the existing folders, or create a new folder.

You can create folders under any of the existing folders as a subfolder. To create a folder:

1. In the Queries window, click the position where you want the folder to be created, and select **File** → **New** → **New Folder**, or right-click and select **New** → **New Folder**, as shown in Figure 6-19. A folder called New Folder is created in the Queries window.

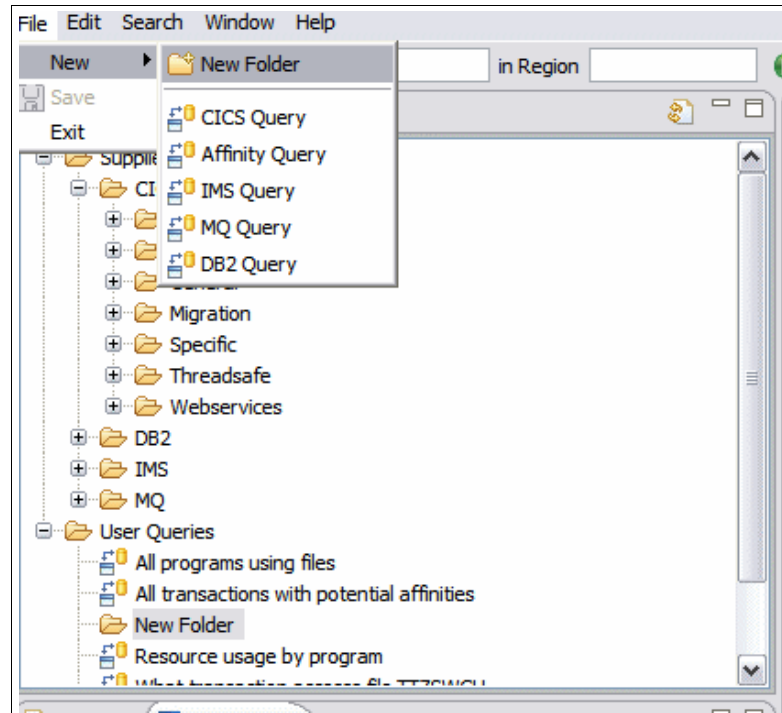


Figure 6-19 Create new folder

2. Change the name New Folder to the required folder name.
3. To delete a query folder, in the Queries window, right-click the query folder name, and click **Delete**. You cannot move an existing folder in the structure.

6.7 Analyzing query and search results

You can further analyze the results of any query or search that are displayed in the CICS IA Explorer windows. You can view the programs and transactions that are using as a resource, and the region in which they are using that resource. You have the option to analyze the resource in the region in which it is being

used. You can analyze the use of each resource across all regions or specific regions.

To further analyze any query or search results:

1. Right-click any resource that is displayed in any window in the CICS IA Explorer.
2. Click **Used By** or **Uses Resources**, as shown in Figure 6-20. Only programs and transactions have the Uses Resources option.

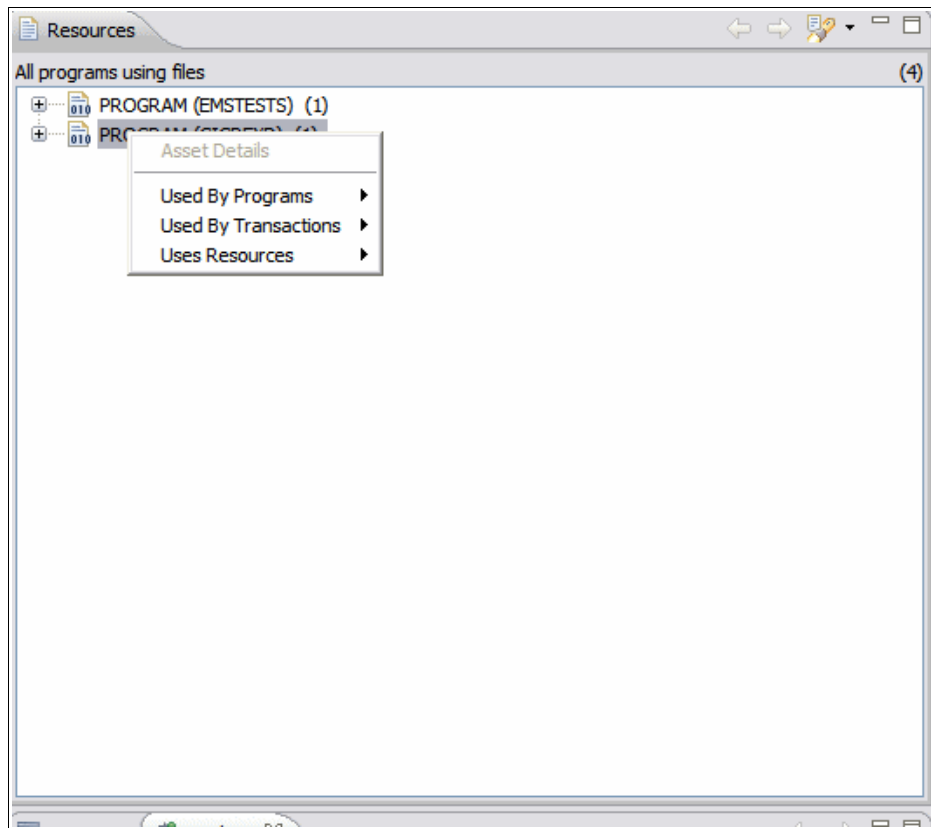


Figure 6-20 Used by and Uses Resources options

3. Click **All Regions** or a specific region that is shown in the menu:
 - The results for the Used By option are displayed in the Used By window.
 - The results for the Uses Resources option are displayed in the Uses windows.

4. Repeat steps 1 to 3 to further analyze the results in the Used By and the Uses windows.

If you select a resource in the Uses window, the lower panel in the Uses window shows the names of the programs that are using that resource. You can navigate the program tree displayed in the lower panel by using the yellow up and down arrows, which move between each resource in order.

5. To recall previous search results in your current CICS IA Explorer session, use the yellow back and forward arrows. The resource currently being analyzed is displayed at the top of the window, as shown in Figure 6-21.

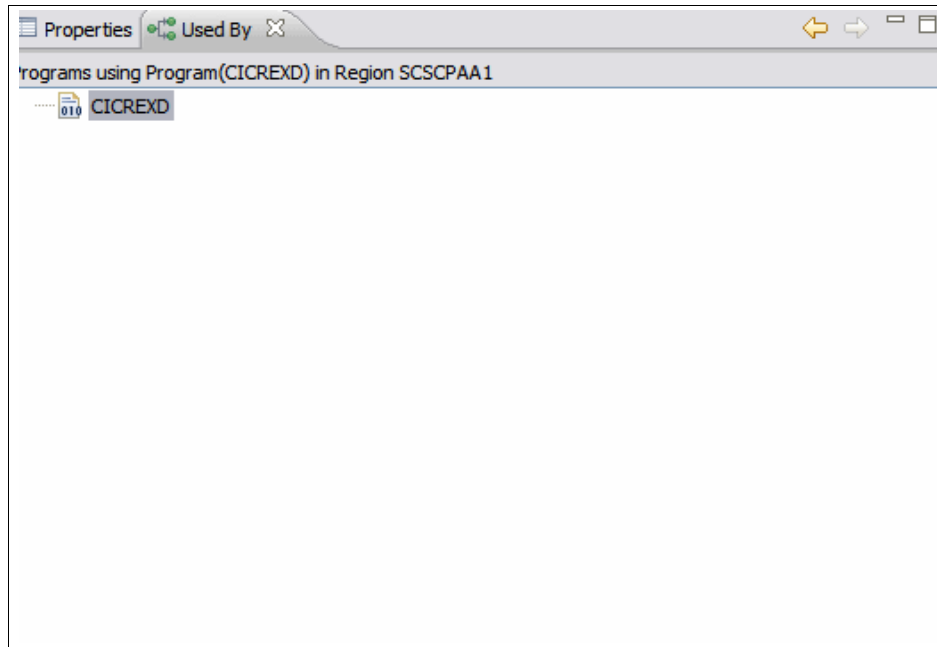


Figure 6-21 After clicking Used By

6.8 Comparing query and search results

When you have a number of query or toolbar search results, you might want to compare two or more results. You can select any number of results in the history using the Compare search results window. You can also compare saved query results.

The Compare search results window relies on complete searches being of the same structure. For large amounts of data, it is more efficient to create a query.

You can compare query and search results using the Compare search results window, which is shown in Figure 6-22:

1. Click the compare torch icon to open the Compare search results window, shown in Figure 6-22. Both panels contain a list of all query results. Select the results in the left panel to Show resources in those results, and compare with the results that you select in the right panel to show resources that are not in those results.

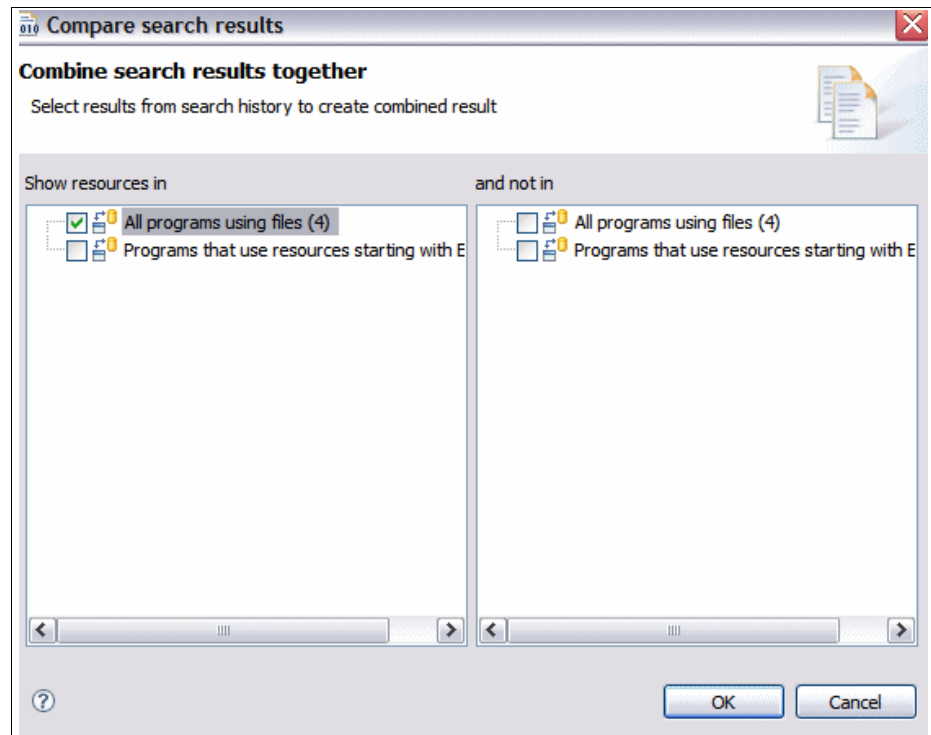


Figure 6-22 Compare search results

If you want to compare saved query results, you must view the saved query results in the Resources window so that the Compare window can include the saved query results in your selection of choices.

2. Click the boxes next to the results that you want to compare.
3. Click **OK**. The Resources window displays the results. The comparison that was run is identified in the grey section of the Resources window. If there are no results, the comparison information is shown but the results window is empty.

4. Optional step: To clear the history of query results from the Compare search results dialog box, click the down arrow next to the torch icon to display the menu items. Click **Clear History**.

6.9 Using WebSphere Studio Asset Analyzer with CICS IA Explorer

If you have WebSphere Studio Asset Analyzer (WSAA) installed, you can access it in the CICS IA Explorer using the Asset Details option.

To use WebSphere Studio Asset Analyzer to analyze query and search results:

1. Click **Window** → **Preferences**.
2. In the Preferences page, on the left navigation pane, click **WSAA**, as shown in Figure 6-23 on page 147.
3. Enter your WSAA server name and port number, and click **Apply**.
4. Click **OK**.
5. Right-click any resource, program, or transaction. If WSAA is relevant for that resource, program, or transaction, the Asset Details option is displayed.
6. Click Asset Details. The WSAA details about the chosen resource are displayed in your external Web browser.

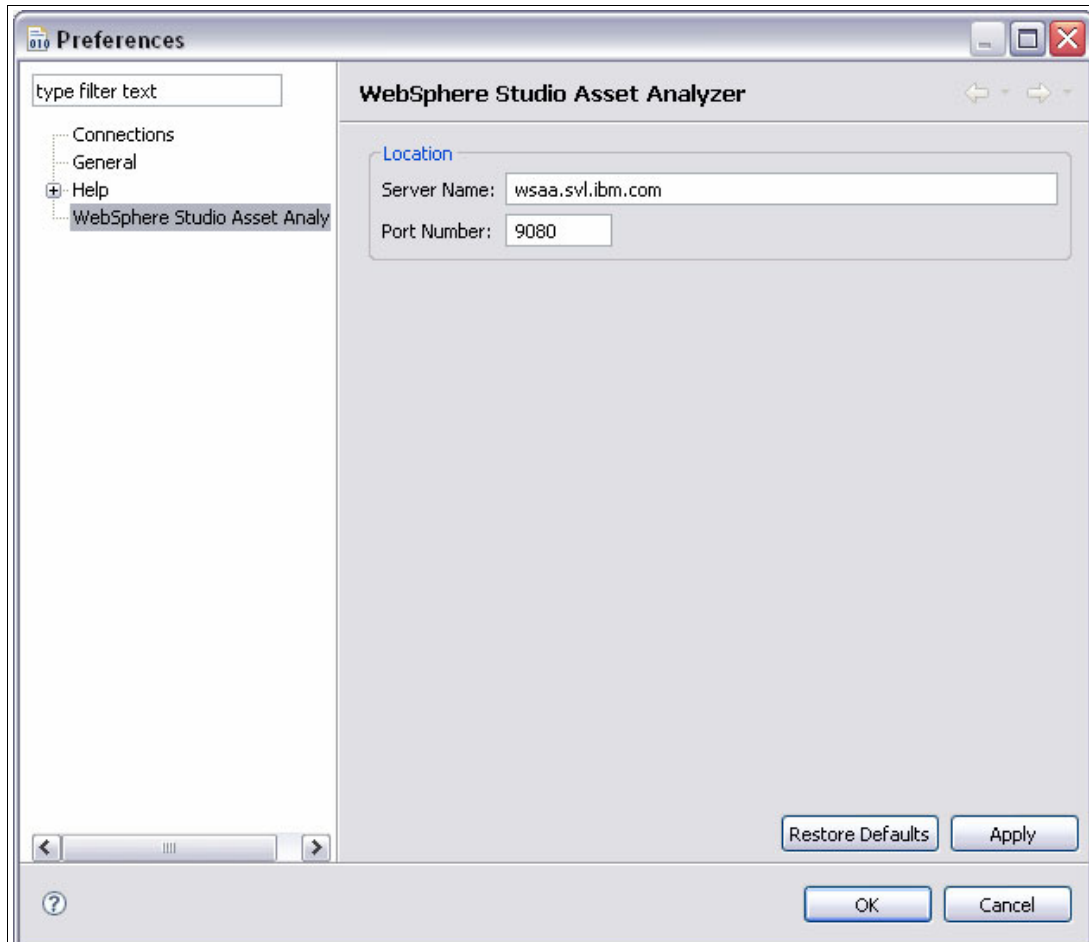


Figure 6-23 WSAA Explorer

7. Right-click a program in the Program folder or a transaction in the Transaction folder.
8. Select the **Asset Details** panel, as shown in Figure 6-24 on page 148, which invokes WebSphere Studio Asset Analyzer.

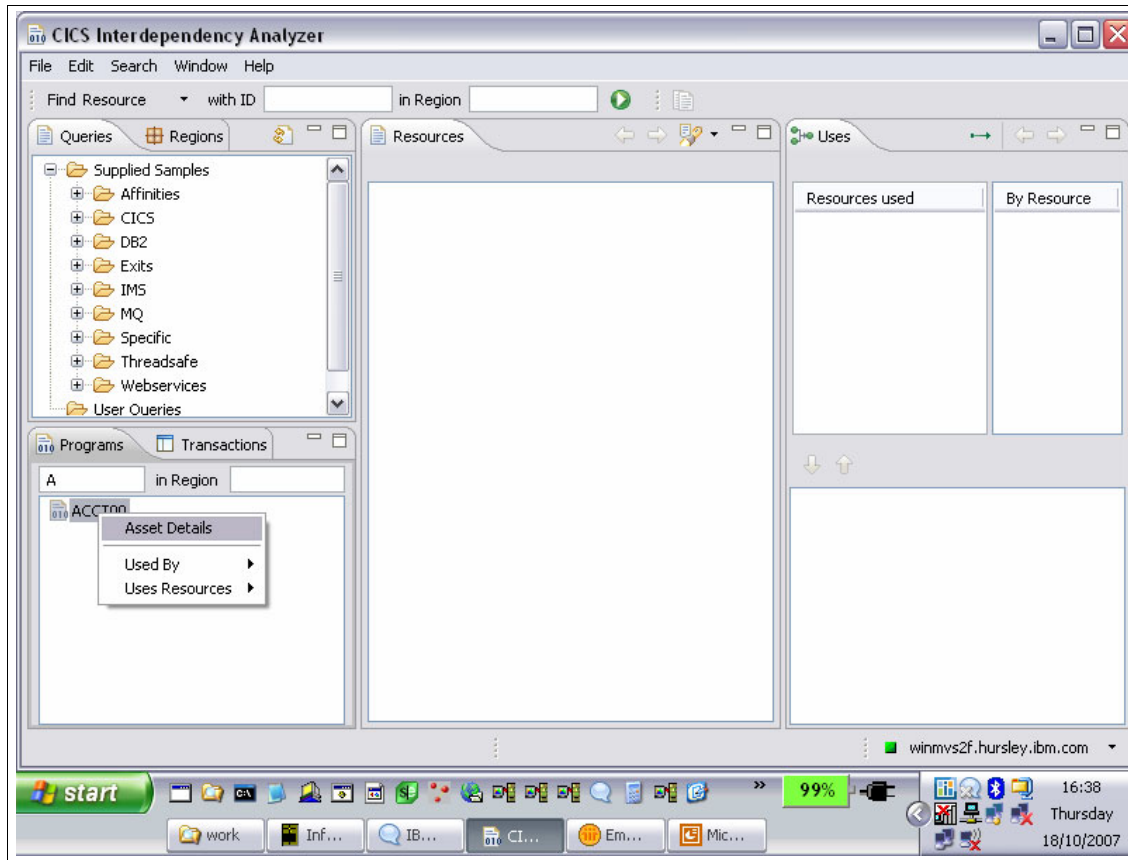


Figure 6-24 CICS IA Explorer - WSAА integration

It passes the program or transaction name and does a search in WSAА. If found, the CICS IA Explorer launches the Program Details view or the CICS Transaction Summary view. See Figure 6-25 on page 149.

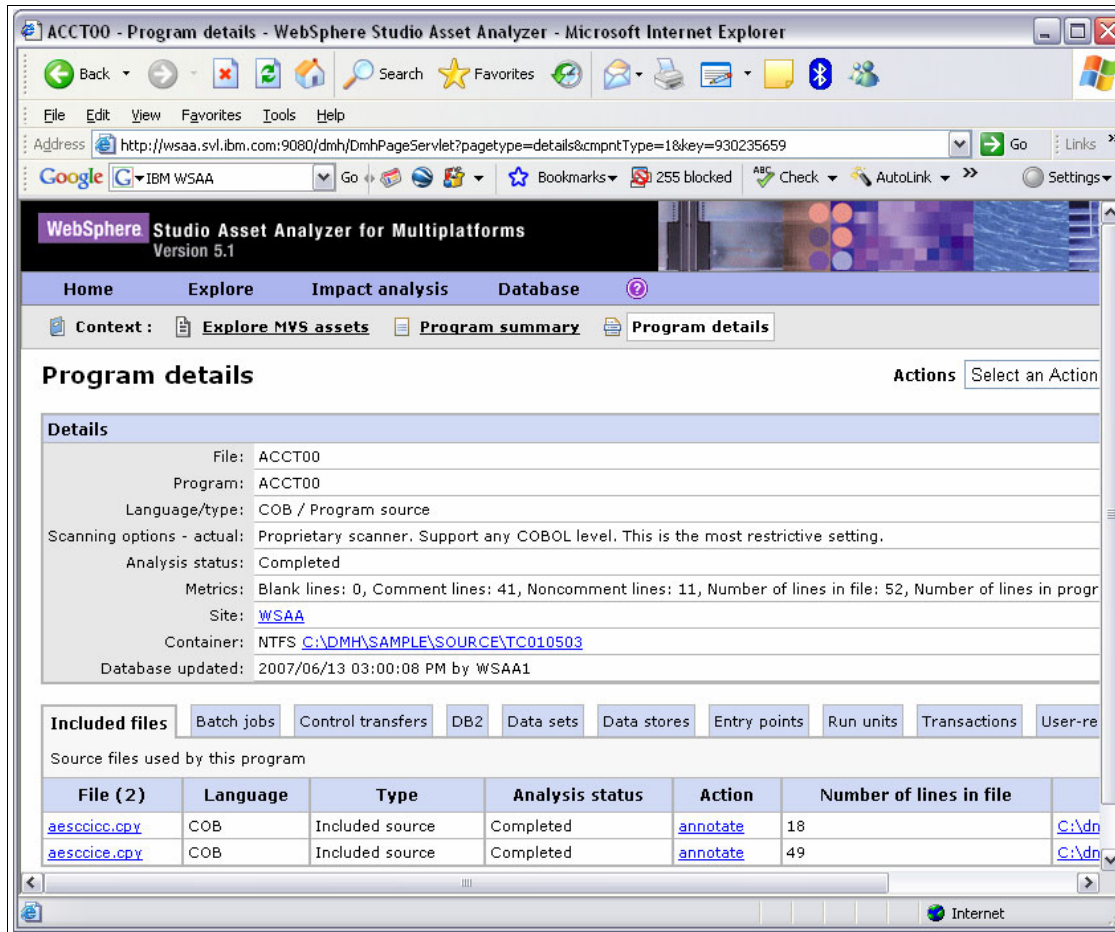


Figure 6-25 CICS IA Explorer - WSAA launch



Sample reports and queries

In this chapter, we look at samples of CICS online queries.

The starter set of CICS online queries is helpful for quick, relatively simple queries. They are in the library *hlq.SCIUSQL*, where *hlq* (high-level qualifier) is the library prefix defined in your installation job stream.

Virtually any possible query can be done by bringing to bear all of the power of SQL on the DB2 database.

Some have found that they already have DB2 information that is related to their CICS applications and resources that can be *joined* to the CICS IA database to provide even further valuable information for their projects.

7.1 Affinities

Inter-transaction affinities require groups of transactions to be run in the same CICS region. Such inter-transaction affinities can be caused by a transaction that leaves data in a place that another transaction (or group of transactions) can only access if they all run in the same CICS region.

Transaction-system affinities require that a particular transaction runs in a particular CICS region. Such transaction-system affinities are caused by interrogating or changing the properties of the CICS region.

To answer questions such as:

- ▶ Which transactions create potentially an inter-transaction affinity?
- ▶ Which transactions might have a transaction-system affinity?

We queried the affinity tables.

These tables are being populated by the Collector. The Collector captures the information while the transaction or program is active in the CICS region(s) that the collector is observing. To get a complete picture, it might therefore be necessary to also query the Load Module Scanner base tables, for example, the CIU_SCAN_DETAIL table.

More information about affinities is in Chapter 9, “Affinities” on page 203.

7.1.1 V_CIU_AFFINITY

This view is a join of CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA.

CIU_AFF_GRP_DATA contains information about affinity transaction groups. An affinity transaction group consists of CICS transactions that have the potential to create an affinity.

CIU_AFF_CMD_DATA contains information about EXEC CICS commands with the potential to create an affinity. Every unique combination of Transaction / Program / EXEC CICS command is recorded. Table 7-1 shows the V_CIU_AFFINITY view.

Table 7-1 The V_CIU_AFFINITY view

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
TRANGROUP	CHAR(10)	Name of the transaction group, for example, TS00000002

Column	Type	Description
AFFTYPE	CHAR(2)	The type of affinity: <ul style="list-style-type: none"> ▶ IT Inter-transaction ▶ TS Transaction-system
GROUPTYPE	CHAR(30)	The group of CICS commands that this transaction group uses. It is one of the following: <ul style="list-style-type: none"> ▶ ADDRESS CWA ▶ CANCEL, DELAY, POST, START group ▶ ENQ and DEQ pair ▶ GETMAIN and FREEMAIN pair ▶ GETMAIN UNMATCHED and FREEMAIN UNMATCHED pair ▶ LOAD and RELEASE pair ▶ LOAD and FREEMAIN pair ▶ LOAD UNMATCHED and FREEMAIN UNMATCHED pair ▶ RETRIEVE ▶ TEMPORARY STORAGE ▶ COLLECT ▶ DISCARD ▶ ENABLE and DISABLE pair
AFFINITY	CHAR(10)	The affinity relation type. It is one of the following: <ul style="list-style-type: none"> ▶ GLOBAL ▶ BACKGROUND ▶ BAPPL ▶ LINK3270 ▶ LUNAME ▶ USERID
AFFWORSENE	CHAR(10)	The relation type from which the affinity has worsened, from one of the following: <ul style="list-style-type: none"> ▶ BACKGROUND ▶ BAPPL ▶ LINK3270 ▶ LUNAME ▶ USERID

Column	Type	Description
LIFETIME	CHAR(10)	The lifetime of the affinity, from one of the following: ▶ ACTIVITY = BTS activity ▶ FACILITY = Link3270 bridge facility ▶ LOGON = Logon ▶ PCONV = Pseudo conversation ▶ PERMANENT = Permanent ▶ PROCESS = BTS process ▶ SIGNON = Signon ▶ SYSTEM = System For an explanation of these lifetime values, see Affinity lifetimes
LIFEWORSEMED	CHAR(10)	The affinity lifetime worsened from one of the following: ▶ ACTIVITY ▶ FACILITY ▶ LOGON ▶ PCONV ▶ PROCESS ▶ SIGNON ▶ SYSTEM
RECOVERY	CHAR(1)	Whether the CICS resource is recoverable: ▶ Y Recoverable ▶ N Not recoverable
RESOURCE	CHAR(255)	The name of the CICS resource, for example, a program name
RESLENGTH	INTEGER	Length of the name of the CICS resource
TYPE	CHAR(8)	The type of the CICS resource, such as TS queue program
TRANCOUNT	INTEGER	The total number of CICS transactions in this affinity group
PROGCOUNT	INTEGER	The total number of CICS programs in this affinity group

Column	Type	Description
BUILD	CHAR(1)	Whether this affinity transaction group is to be included in a combined affinity-transaction-group definition, created by the CICS IA Builder: <ul style="list-style-type: none"> ▶ Y: This affinity transaction-group is to be included ▶ N: This affinity transaction group is not to be included
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program
OFFSET	CHAR(8)	Offset, from the start of the program at which this command occurs.
COMMAND	CHAR(8)	EXEC CICS command.
RESTYPE	CHAR(8)	Resource type, for example, program
TERMINAL	CHAR(1)	Whether there is a terminal associated with the transaction: <ul style="list-style-type: none"> ▶ Y: Terminal transaction ▶ N: Non terminal transaction
BTS	CHAR(1)	Whether there is a BTS task: <ul style="list-style-type: none"> ▶ Y: BTS task ▶ N: Non BTS task
LINK3270	CHAR(1)	Whether there is a LINK3270 transaction: <ul style="list-style-type: none"> ▶ Y: LINK3270 transaction ▶ N: Non LINK3270 transaction

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples:

- ▶ Show all transactions that potentially have an inter-transaction affinity:

```

SELECT TRANSID,
       AFFINITY,
       AFFWORSENER,
       LIFETIME,
       LIFEWORSENER,
       RESOURCE,
       TYPE
FROM V_CIU_AFFINITY READONLY
ORDER BY TRANSID;

```

- ▶ Show all programs that contain the CICS command “ADDRESS CWA”:

```
SELECT PROGRAM,
       TRANSID
FROM V_CIU_AFFINITY READONLY
WHERE GROUPTYPE = 'ADDRESS CWA'
ORDER BY PROGRAM;
```

7.2 Dependencies

Dependency tables can be used to answer questions, such as:

- ▶ Which resources are used by a specific transaction?
- ▶ Which programs are running on an L8 TCB?
- ▶ Which Files are used (directly and indirectly) by a specific transaction?

We queried some of the dependency base tables. The Collector is populating these tables and captures the information while the transaction or program is active in the CICS region(s) that the collector is observing. To get a complete picture, it might therefore be necessary to also query the Load Module Scanner base tables, for example, the CIU_CSECT_INFO table.

7.2.1 CIU_CICS_DATA

Table 7-2 contains information about the function and the program that issued the EXEC CICS command (function).

Table 7-2 The CIU_CICS_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program

Column	Type	Description
FUNCTION	CHAR(8)	EXEC CICS command name, such as READ, WRITEQ: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For information about FUNCTION and TYPE values, see Table 7-3 on page 158
TYPE	CHAR(8)	The resource type, such as TS, program: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For information about FUNCTION and TYPE values, see Table 7-3 on page 158
OBJECT	CHAR(255)	Resource name
OBJLENGTH	INTEGER	Length of resource name in OBJECT column.
RMTSYSID	CHAR(4)	Remote SYSID, where relevant
RMTNAME	CHAR(8)	Name by which the resource is known in the remote region
TERMTRAN	CHAR(1)	Whether there is a terminal associated with the transaction: <ul style="list-style-type: none"> ▶ Y: Terminal transaction ▶ N: Non terminal transaction
TCBMODE	CHAR(2)	CICSTCB mode at the time of the EXEC call. For example QR or L8
AFFINITY	CHAR(1)	Whether the EXEC CICS command could cause an affinity: <ul style="list-style-type: none"> ▶ Yes ▶ No
OFFSET	CHAR(8)	OFFSET of the EXEC call into the program
PROGLEN	CHAR(8)	Size of program
COMMAREA	CHAR(1)	Whether there is a commarea associated with the transaction: <ul style="list-style-type: none"> ▶ Yes ▶ No

Column	Type	Description
CHANNEL	CHAR(1)	Whether there is a channel associated with the transaction: <ul style="list-style-type: none"> ▶ Yes ▶ No
USECOUNT	INTEGER	Number of times this EXEC CICS call is seen within the program
FIRST_RUN	TIMESTAMP	Time of first occurrence
LAST_RUN	TIMESTAMP	Time of latest occurrence

TYPE and FUNCTION mapping - CICS commands

Table 7-3 lists the possible values for resource TYPE and the FUNCTION associated with that type. Using these fields, you can write SQL queries to generate a report to suit your needs.

Table 7-3 Resource types and associated functions

Resource type	Function	CICS command
(blank)	ADDRESS ALLOCATE ASSIGN CONVERSE FREE RECEIVE ROUTE SEND SIGNOFF SIGNON	ADDRESS ALLOCATE ASSIGN CONVERSE FREE RECEIVE ROUTE SEND SIGNON SIGNOFF
ABEND	ISSUE	ISSUE ABEND
BRFACIL	INQ NEXT INQUIRE SET	INQUIRE BRFACILITY NEXT INQUIRE BRFACILITY SET BRFACILITY
CHANNEL	DEL CNTR GET CNTR LINK MOV CNTR PUT CNTR RETURN XCTL START	DELETE CONTAINER CHANNEL GET CONTAINER CHANNEL LINK PROGRAM CHANNEL MOVE CONTAINER CHANNEL PUT CONTAINER CHANNEL RETURN CHANNEL XCTL PROGRAM CHANNEL START CHANNEL
CONFRMTN	ISSUE	ISSUE CONFIRMATION

Resource type	Function	CICS command
CONTAINR	DELETE GET MOVE PUT	DELETE CONTAINER GET CONTAINER MOVE CONTAINER PUT CONTAINER
COPY	ISSUE	ISSUE COPY
CORBASRV	CREATE DISCARD INQ NEXT INQUIRE PERFORM SET	CREATE CORBASERVER DISCARD CORBASERVER INQUIRE CORBASERVER NEXT INQUIRE CORBASERVER PERFORM CORBASERVER SET CORBASERVER
COUNTER	DEFINE DEFINE DELETE DELETE GET GET QUERY QUERY REWIND REWIND UPDATE UPDATE	DEFINE COUNTER DEFINE DCOUNTER DELETE COUNTER DELETE DCOUNTER GET COUNTER GET DCOUNTER QUERY COUNTER QUERY DCOUNTER REWIND COUNTER REWIND DCOUNTER UPDATE COUNTER UPDATE DCOUNTER
DB2ENTRY	CREATE DISCARD INQ NEXT INQUIRE SET	CREATE DB2ENTRY DISCARD DB2ENTRY INQUIRE DB2ENTRY NEXT INQUIRE DB2ENTRY SET DB2ENTRY
DB2TRAN	CREATE DISCARD INQ NEXT INQUIRE SET	CREATE DB2TRAN DISCARD DB2TRAN INQUIRE DB2TRAN NEXT INQUIRE DB2TRAN SET DB2TRAN
DISCONNT	ISSUE	ISSUE DISCONNECT
DJAR	CREATE DISCARD INQ NEXT INQUIRE PERFORM	CREATE DJAR DISCARD DJAR INQUIRE DJAR NEXT INQUIRE DJAR PERFORM DJAR

Resource type	Function	CICS command
DOCTEMP	CREATE DISCARD INQ NEXT INQUIRE	CREATE DOCTEMPLATE DISCARD DOCTEMPLATE INQUIRE DOCTEMPLATE NEXT INQUIRE DOCTEMPLATE
ENQNAME	DEQ DEQSYS ENQ ENQSYS	DEQ DEQ (scope is sysplex-wide) ENQ ENQ (scope is sysplex-wide)
ERROR	ISSUE	ISSUE ERROR
EXIT	CALL DISABLE ENABLE EXTRACT	(CALL to TRUE) DISABLE PROGRAM ENABLE PROGRAM EXTRACT EXIT
FEPI	EXTRACTF EXTRACTS FREE ISSUE RECEIVE RECEIVE REQTKT SEND SEND START	FEPI EXTRACT FIELD FEPI EXTRACT STNS FEPI FREE FEPI ISSUE FEPI RECEIVE DATASTREAM FEPI RECEIVE FORMATED FEPI REQUEST PASSTICKET FEPI SEND DATASTREAM FEPI SEND FORMATTED FEPI START
FEPINODE	INQ CONN INQ NODE SET CONN SET NODE	FEPI INQUIRE TARGET FEPI INQUIRE NODE FEPI SET CONNECTION FEPI SET NODE
FEPIPOOL	ALLOCATE CONVERSE CONVERSE EXTRACTC ADD POOL DEL POOL DISCPOOL INQ POOL INSTPOOL SET POOL	FEPI ALLOCATE POOL FEPI CONVERSE DATASTREAM FEPI CONVERSE FORMATTED FEPI EXTRACT CONV FEPI ADD POOL FEPI DELETE POOL FEPI DISCARD POOL FEPI INQUIRE POOL FEPI INSTALL POOL FEPI SET POOL
FEPISET	DISCPSET INQ PSET INSTPSET	FEPI DISCARD POOL FEPI INQUIRE PROPERTYSET FEPI INSTALL PROPERTYSET

Resource type	Function	CICS command
FEPITGT	INQ TRGT SET TRGT	FEPI INQUIRE TARGET FEPI SET CONNECTION
FILE	DELETE ENDBR READ READ UPD READNEXT READPREV RESETBR REWRITE STARTBR UNLOCK WRITE CREATE DISCARD INQ NEXT INQUIRE SET	DELETE ENDBR READ READ UPDATE READNEXT READPREV RESETBR REWRITE STARTBR UNLOCK WRITE CREATE FILE DISCARD FILE INQUIRE FILE NEXT INQUIRE FILE SET FILE
HANDLE	PUSH POP	PUSH HANDLE POP HANDLE
IPCONN	CREATE DISCARD INQUIRE SET	CREATE IPCONN DISCARD IPCONN INQUIRE IPCONN SET IPCONN
JOURNAL	WAIT WAIT WRITE WRITE DISCARD INQ NEXT INQ NEXT INQUIRE INQUIRE SET SET	WAIT JOURNALNAME WAIT JOURNALNUM WRITE JOURNALNAME WRITE JOURNALNUM DISCARD JOURNALNAME INQUIRE JOURNALNAME NEXT INQUIRE JOURNALNUM NEXT INQUIRE JOURNALNAME INQUIRE JOURNALNUM SET JOURNALNAME SET JOURNALNUM
JVMPROF	INQ NEXT INQUIRE	INQUIRE JVMPROFILE NEXT INQUIRE JVMPROFILE
LIBRARY	CREATE DISCARD INQUIRE SET	CREATE LIBRARY DISCARD LIBRARY INQUIRE LIBRARY SET LIBRARY

Resource type	Function	CICS command
MAP	PURGE RECEIVE SEND	PURGE MESSAGE RECEIVE MAP SEND MAP
MAPSET	RECV MAP SEND MAP	RECEIVE MAP MAPSET SEND MAP MAPSET
PASS	ISSUE	ISSUE PASS
PIPELINE	CREATE DISCARD INQ NEXT INQUIRE PERFORM SET	CREATE PIPELINE DISCARD PIPELINE INQUIRE PIPELINE NEXT INQUIRE PIPELINE PERFORM PIPELINE SET PIPELINE
POOL	DEF CTR DEF DCTR DEL CTR DEL DCTR GET CTR GET DCTR QRY CTR QRY DCTR REW CTR REW DCTR UPD CTR UPD DCTR	DEFINE COUNTER POOL DEFINE DCOUNTER POOL DELETE COUNTER POOL DELETE DCOUNTER POOL GET COUNTER POOL GET DCOUNTER POOL QUERY COUNTER POOL QUERY DCOUNTER POOL REWIND COUNTER POOL REWIND DCOUNTER POOL UPDATE COUNTER POOL UPDATE DCOUNTER POOL
PROCESS	EXTRACT CONNECT	EXTRACT PROCESS CONNECT PROCESS
PROGRAM	CALL HANDABND LINK LOAD XCTL CREATE DISCARD INQ NEXT INQUIRE SET	Dynamic program call HANDLE ABEND LINK LOAD XCTL CREATE PROGRAM DISCARD PROGRAM INQUIRE PROGRAM NEXT INQUIRE PROGRAM SET PROGRAM
RESET	ISSUE	ISSUE RESET
SIGNAL	ISSUE	ISSUE SIGNAL
STORAGE	FREEMAIN GETMAIN	FREEMAIN GETMAIN

Resource type	Function	CICS command
STORSHR	GETMAIN	GETMAIN SHARED
TCPIPSRV	CREATE DISCARD INQ NEXT INQUIRE SET	CREATE TCPIP SERVICE DISCARD TCPIP SERVICE INQUIRE TCPIP SERVICE NEXT INQUIRE TCPIP SERVICE SET TCPIP SERVICE
TD	DELETEQ READQ WRITEQ CREATE DISCARD INQ NEXT INQUIRE SET	DELETEQ TD READQ TD WRITEQ TD CREATE TDQUEUE DISCARD TDQUEUE INQUIRE TDQUEUE NEXT INQUIRE TDQUEUE SET TDQUEUE
TERMINAL	WAIT	WAIT TERMINAL
TEXT	SEND	SEND TEXT
TRANSID	CREATE DISCARD INQ NEXT INQUIRE SET RETURN START START START STARTREQ	CREATE TRANSACTION DISCARD TRANSACTION INQUIRE TRANSACTION NEXT INQUIRE TRANSACTION SET TRANSACTION RETURN START START ATTACH START BREXIT START REQID
TS	DELETEQ READQ WRITEQ INQ NEXT INQ NEXT INQUIRE INQUIRE SET SET	DELETEQ TS READQ TS WRITEQ TS INQUIRE TSQNAME NEXT INQUIRE TSQUEUE NEXT INQUIRE TSQNAME INQUIRE TSQUEUE SET TSQNAME SET TSQUEUE
TSAU	DELETEQ READQ WRITEQ	DELETEQ TS (auxiliary storage) READQ TS (auxiliary storage) WRITEQ TS (auxiliary storage)

Resource type	Function	CICS command
TSMODEL	CREATE DISCARD INQUIRE	CREATE TSMODEL DISCARD TSMODEL INQUIRE TSMODEL
TSPOOL	INQUIRE	INQUIRE TSPOOL
TSSHR	DELETEQ READQ WRITEQ	DELETEQ TS (shared) READQ TS (shared) WRITEQ TS (shared)
UOW	ROLLBACK SYNC	SYNCPPOINT ROLLBACK SYNCPPOINT
URIMAP	CREATE DISCARD INQ NEXT INQUIRE SET	CREATE URIMAP DISCARD URIMAP INQUIRE URIMAP NEXT INQUIRE URIMAP SET URIMAP
WEB	ENDBR EXTRACT READ READNEXT RECEIVE RETRIEVE SEND STARTBR WRITE	WEB ENDBROWSE WEB EXTRACT WEB READ WEB READNEXT WEB RECEIVE WEB RETRIEVE WEB SEND WEB STARTBROWSE WEB WRITE HTTPHEADER
WEBSRV	INQ NEXT INQUIRE INVOKE SET	INQUIRE WEBSERVICE NEXT INQUIRE WEBSERVICE INVOKE WEBSERVICE SET WEBSERVICE

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples:

- Show all of the resources that the transaction MAIL used:

```
SELECT PROGRAM,
        FUNCTION,
        TYPE,
        OBJECT
FROM CIU_CICS_DATA READONLY
WHERE TRANSID = 'MAIL'
ORDER BY OBJECT, FUNCTION;
```

- ▶ Show all programs that the transaction MAIL invoked:

```
SELECT DISTINCT PROGRAM
FROM CIU_CICS_DATA READONLY
WHERE TRANSID = 'MAIL'
ORDER BY PROGRAM;
```

- ▶ Show all of the transactions that were run only once:

```
SELECT DISTINCT TRANSID
FROM CIU_CICS_DATA
WHERE FIRST_RUN = LAST_RUN;
```

7.2.2 V_CIU_CICS_INDS

The V_CIU_CICS_INDS view is a join of tables: CIU_CICS_CHAIN and CIU_CICS_DATA.

CIU_CICS_DATA contains information about the function and the program that issued the EXEC CICS command (function).

CIU_CICS_CHAIN contains the name of the starting transaction and the name of the connected transaction that it starts or with which it holds a DTP conversation.

A selection on FRONT_TXN in view V_CIU_CICS_INDS now shows all of the resources, direct and indirect, that this transaction can use. A selection on a specific resource reports all of the transactions that might be affected, directly and indirectly, if the resource is unavailable.

Table CIU_CICS_CHAIN consists of 2 columns:

- ▶ FRONT_TXN for the starting transaction
- ▶ BACK_TXN for the connected transaction

The chain of transactions that got started or with which a Distributed Transaction Processing conversation was held is reflected in this table. Example 7-1 shows the structure of CIU_CICS_CHAIN.

Example 7-1 Structure of CIU_CICS_CHAIN

-
- ▶ TRN1 connects to TRN2
 - ▶ TRN2 connects to TRN3

To capture the indirect dependencies, three rows in CIU_CICS_CHAIN are needed.

FRONT_TXN	BACK_TXN
TRN1	TRN2

FRONT_TXN	BACK_TXN
TRN2	TRN3
TRN1	TRN3

To complete the picture, a further record is needed to include also immediate dependencies.

FRONT_TXN	BACK_TXN
TRN1	TRN1

Table 7-4 shows the V_CIU_CICS_INDS view.

Table 7-4 The V_CIU_CICS_INDS view

Column	Type	Description
FRONT_TXN	CHAR(4)	Transaction to be analyzed
BACK_TXN	CHAR(4)	Dependent transaction for join to TRANSID
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program
FUNCTION	CHAR(8)	EXEC CICS command name, such as READ, WRITEQ: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For information about FUNCTION and TYPE values, see Table 7-3 on page 158
TYPE	CHAR(8)	The resource type, such as TS, program: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For information about FUNCTION and TYPE values, see Table 7-3 on page 158

Column	Type	Description
OBJECT	CHAR(255)	Resource name
RMTSYSID	CHAR(4)	Remote SYSID, where relevant
TERMTRAN	CHAR(1)	Whether there is a terminal associated with the transaction: ▶ Y: Terminal transaction ▶ N: Non terminal transaction
USECOUNT	INTEGER	Number of times this EXEC CICS call is seen within the program
LAST_RUN	TIMESTAMP	Time of latest occurrence

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples:

- ▶ Show all files that are used (direct and indirect) for transaction TTAC:

```
SELECT BACK_TXN,
       PROGRAM,
       FUNCTION,
       OBJECT
FROM V_CIU_CICS_INDS READONLY
WHERE FRONT_TXN = 'TTAC' AND
       TYPE = 'FILE'
ORDER BY OBJECT,
       PROGRAM;
```

- ▶ Show all of the resources that the program TTZ3SWCF uses:

```
SELECT DISTINCT PROGRAM,
               TRANSID,
               FUNCTION,
               TYPE,
               OBJECT
FROM V_CIU_CICS_INDS READONLY
WHERE PROGRAM = 'TTZ3SWCF'
ORDER BY OBJECT,
       TYPE,
       FUNCTION;
```

7.2.3 CIU_DB2_DATA

Table 7-5 on page 168 contains information about the EXEC SQL command and the program that issues the EXEC SQL command.

Table 7-5 The CIU_DB2_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
DB2ID	CHAR(4)	DB2 subsystem ID
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program
PLAN	CHAR(8)	DB2 planname
RESTYPE	CHAR(10)	DB2 resource type. For information about FUNCTION and RESTYPE, values see Table 7-6 on page 170
RESNAME	CHAR(40)	DB2 resource name. It is possible that this column might not contain the actual name of the DB2 resource being used; instead, it might, for example, contain the name of a variable. In this case, use the values of the PROGRAM, SECTION, and STATEMENT columns to look up, in the DB2 SYSIBM.SYSPACKSTMT or SYSIBM.SYSSTMT table, the DB2 resource name. The views V_CIU_DB2_RES for SYSIBM.SYSPACKSTMT and V_CIU_DB2_RES2, for SYSIBM.SYSSTMT are provided to help you do this
FUNCTION	CHAR(24)	EXEC SQL command name, such as CREATE, UPDATE: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For information about FUNCTION and RESTYPE values, see Table 7-6 on page 170
SECTION	SMALLINT	The section number, in the source code of the CICS program, at which the DB2 command is issued
STATEMENT	SMALLINT	The pre-compiler statement number, in the source code of the CICS program, at which the DB2 command is issued

Column	Type	Description
TERMTRAN	CHAR(1)	Whether there is a terminal associated with the transaction: <ul style="list-style-type: none"> ▶ Y: Terminal transaction ▶ N: Non-terminal transaction
TCBMODE	CHAR(2)	CICS TCB in which the application is running, for example, QR or L8
OFFSET	CHAR(8)	The offset of the command from the start of the program
PROGLEN	CHAR(8)	Length of CICS program. Used for program versioning
USECOUNT	INTEGER	Number of occurrences
FIRST_RUN	TIMESTAMP	Time of first occurrence
LAST_RUN	TIMESTAMP	Time of latest occurrence

RESTYPE and FUNCTION values in DB2 queries

Table 7-6 on page 170 lists the possible combinations of TYPE and FUNCTION values in DB2 queries.

Table 7-6 Resource types and possible function values in DB2 Queries

Resource type	Function
(blank)	EXPLAIN SET CURRENT SQLID SET CURRENT PACKAGESET SET CURRENT DEGREE SET HOST VAR INTOPEN GRANT REVOKE REMOTE SQL ROLLBACK LOCK COMMIT COMMENT ON LABEL ON CONNECT TO CONNECT RESET CONNECT IMPLICIT CONNECT TYPE2 CONNECT TO TYPE2 CONNECT RESET TYPE2 CONNECT SET CONNECTION RELEASE LOCATION RELEASE CURRENT RELEASE ALL RELEASE ALL SQL RELEASE ALL PRIVATE SET CURRENT RULES CALL STATEMENT DESCRIBE PROCEDURE ASSOCIATE LOCATORS FETCH ALLOC CURSOR CLOSE ALLOC CURSOR DESCRIBE ALLOC CURSOR DESCRIBE INPUT SET SPECIAL REGISTER
ALIAS	CREATE ALIAS DROP ALIAS
CURSOR	OPEN FETCH CLOSE ALLOCATE CURSOR

Resource type	Function
DATABASE	CREATE DATABASE DROP DATABASE ALTER DATABASE
Dynamic (mixed case!)	EXECUTE IMMEDIATE
INDEX	CREATE INDEX DROP INDEX ALTER INDEX
PACKAGE	DROP PACKAGE/PROGRAM
STATEMENT	PREPARE EXECUTE DESCRIBE
STOGROUP	CREATE STOGROUP DROP STOGROUP ALTER STOGROUP
SYNONYM	CREATE SYNONYM DROP SYNONYM
TABLE	SELECT INSERT DELETE UPDATE RENAME TABLE CREATE TABLE ALTER TABLE DROP TABLE
TABLESPACE	CREATE TABLESPACE DROP TABLESPACE ALTER TABLESPACE
VIEW	CREATE VIEW DROP VIEW

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The samples show all transactions that use DB2 resources:

```
SELECT DISTINCT TRANSID,
           PROGRAM,
           PLAN
FROM CIU_DB2_DATA READONLY
ORDER BY TRANSID,
```

PROGRAM;

7.2.4 CIU_IMS_DATA

CIU_IMS_DATA contains information about the DLI command and the program that issued the command.

Table 7-7 The CIU_IMS_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program
CALLTYPE	CHAR(4)	Reserved
FUNCTION	CHAR(12)	DLI command. For more information about FUNCTION and TYPE values see Table 7-8 on page 173
TYPE	CHAR(4)	PSB or PCB For more information about FUNCTION and TYPE values see Table 7-8 on page 173
OBJECT	CHAR(8)	PSB or PCB name
TERMTRAN	CHAR(1)	Whether there is a terminal associated with the transaction: ▶ Y: Terminal transaction. ▶ N: Non-terminal transaction.
TCBMODE	CHAR(2)	CICS TCB in which the application is running, for example, QR or L8
OFFSET	CHAR(8)	The offset of the command from the start of the program
PROGLEN	CHAR(8)	Length of the CICS program. Used for program versioning
USECOUNT	INTEGER	Number of occurrences
FIRST_RUN	TIMESTAMP	Time of first occurrence
LAST_RUN	TIMESTAMP	Time of latest occurrence

TYPE and FUNCTION values in IMS Queries

Table 7-8 lists the possible combinations of TYPE and FUNCTION values in IMS queries.

Table 7-8 Resource types and possible function values in IMS Queries

Resource type	Function
PCB	DELETE GET NEXT GET NEXT INP GET UNIQUE INSERT REPLACE
PSB	SCHEDULE

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The samples show all of the transactions that use IMS resources:

```
SELECT DISTINCT TRANSID,  
        PROGRAM,  
        FUNCTION,  
        OBJECT,  
        TYPE  
FROM CIU_IMS_DATA READONLY  
ORDER BY TRANSID,  
        OBJECT,  
        FUNCTION;
```

7.2.5 CIU_MQ_DATA

Table 7-9 contains information about the MQ command and the program that issued the command.

Table 7-9 The CIU_MQ_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program

Column	Type	Description
FUNCTION	CHAR(8)	MQ command, such as MQGET, MQPUT: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For more information about FUNCTION and TYPE values, see Table 7-10 on page 175
TYPE	CHAR(8)	Resource type: <ul style="list-style-type: none"> ▶ If the database update program does not recognize the EIBCODE in the dependency data, both the FUNCTION and TYPE columns contain ???????? ▶ For more information about FUNCTION and TYPE values, see Table 7-10 on page 175
OBJECT	CHAR(48)	Resource name
TERMTRAN	CHAR(1)	Whether there is a terminal associated with the transaction: <ul style="list-style-type: none"> ▶ Y: Terminal transaction ▶ N: Non-terminal transaction
TCBMODE	CHAR(2)	CICS TCB in which the application is running, for example, QR or L8
OFFSET	CHAR(8)	The offset of the command from the start of the program
PROGLEN	CHAR(8)	Length of CICS program. Used for program versioning.
MQFIQ	CHAR(1)	Whether the MQOO_FAIL_IF_QUIESCING option is set for the FUNCTION: <ul style="list-style-type: none"> ▶ Yes ▶ No
MQBOO	CHAR(1)	Whether the MQOO_BIND_ON_OPEN option is set for the FUNCTION: <ul style="list-style-type: none"> ▶ Yes ▶ No
USECOUNT	INTEGER	Number of occurrences
FIRST_RUN	TIMESTAMP	Time of first occurrence
LAST_RUN	TIMESTAMP	Time of latest occurrence

TYPE and FUNCTION values in MQ Queries

Table 7-10 lists the possible combinations of TYPE and FUNCTION values in MQ queries.

Table 7-10 Resource types and possible function values in MQ Queries

Resource type	Function
QUEUE	CLOSE GET OPEN PUT PUT1

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples show all of the transactions that use the MQPUT1 command:

```
SELECT DISTINCT TRANSID,  
PROGRAM,  
FUNCTION,  
OBJECT  
FROM CIU_MQ_DATA READONLY  
WHERE FUNCTION = 'PUT1'  
ORDER BY TRANSID,  
OBJECT,  
FUNCTION;
```

7.3 Resources

In this section, we review some of the resource tables, which the Collector populates.

For details about all of the resource tables that we do not cover in this chapter, such as programs(CIU_PROGRAM_DETAIL table), TDQueues(CIU_TDQUEUE_DETAIL table), TSQueues(CIU_TSQUEUE_DETAIL table), see the *CICS Interdependency Analyzer for z/OS User's Guide and Reference*, SC34-6790-01.

7.3.1 CIU_EXIT_INFO

Table 7-11 on page 176 contains information about every CICS GLUE and TRUE invoked by at least one transaction recorded by the collector.

Note: Information is gathered only if the Exits field on the CICS Resource Options panel (CIU240) is set to **Y**.

Table 7-11 The CIU_EXIT_INFO table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local CICS region.
EXIT_PROGRAM	CHAR(8)	Name of the exit program.
EXIT_NAME	CHAR(8)	Name of the exit.
EXIT_POINT	CHAR(8)	The name of the entry point that is associated with the exit. GLUEs only.
EXIT_TYPE	CHAR(4)	Type of exit. Values are GLUE or TRUE.
FIRST_RUN	TIMESTAMP	Time of first occurrence.
LAST_RUN	TIMESTAMP	Time of latest observation.

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The samples show all of the exits that CICS IA uses:

```
SELECT DISTINCT EXIT_POINT,  
               EXIT_PROGRAM  
FROM CIU_EXIT_INFO READONLY  
WHERE EXIT_PROGRAM LIKE 'CIU%'  
ORDER BY EXIT_POINT;
```

7.3.2 CIU_FILE_DETAIL

Table 7-12 on page 177 contains information about every CICS file that is referenced in a transaction that the Collector records.

Note: Information is gathered only if the Files field on the CICS Resource Options panel (CIU240) is set to **D**.

Table 7-12 The CIU_FILE_DETAIL table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local CICS region
FILE_NAME	CHAR(8)	Name of the file resource
ACCESSMETHOD	CHAR(8)	Access Method of the file. Values are: BDAM, REMOTE, and VSAM
BASEDSNAME	CHAR(44)	The base name of the file resource
BLOCKFORMAT	CHAR(10)	Whether records in the file are blocked or unblocked. The values are: BLOCKED and UNBLOCKED
BLOCKKEYLEN	INTEGER	Physical block key length for the file
BLOCKSIZE	INTEGER	The length in bytes for the block size of the file
CFDTPOOL	CHAR(8)	Name of the coupling facility data table pool
DISPOSITION	CHAR(8)	The disposition option for the file. The values are: OLD and SHARE
DSNAME	CHAR(44)	The data set name of the file resource
JOURNALNUM	INTEGER	The number of the journal to which CICS writes the information that is required for autojournaling
KEYLENGTH	INTEGER	The length of the record key for the file
KEYPOSITION	INTEGER	The starting position of the Key field in each record relative to the beginning of the record
LOADTYPE	CHAR(10)	The load type for a coupling facility data table. The values are: LOAD, NOLOAD, and NOTAPPLIC
LSRPOOLID	INTEGER	The number of VSAM LSR pools that are associated with this file
MAXNUMRECS	INTEGER	The maximum number of records that the file can hold

Column	Type	Description
OBJECT	CHAR(8)	Whether the file is associated with a data set or a VSAM path that links an alternate index to its base cluster. The values are: <ul style="list-style-type: none"> ▶ BASE the file is associated with a data set that is a VSAM base ▶ PATH the file is associated with a path
RBATYPE	CHAR(12)	Identifies if a VSAM file uses extended addressing. The values are: EXTENDED, NOTEXTENDED, and NOTAPPLIC
RECORDFORMAT	CHAR(10)	Identifies the format of the records on the file. The values are: FIXED, VARIABLE, or UNDEFINED
RECORDSIZE	INTEGER	The actual size of a fixed-length record or the maximum size of a variable-length record
RECOVSTATUS	CHAR(14)	Indicates whether the file is recoverable. The values are: NOTRECOVERABLE or RECOVERABLE
RELTYPE	CHAR(10)	Indicates whether relative or absolute addressing is used to access the file and, if relative, the type of relative addressing. The values are: <ul style="list-style-type: none"> ▶ BLK Relative block addressing ▶ DEC Zoned decimal format ▶ HEX Hexadecimal relative track format ▶ NOTAPPLIC Absolute addressing is being used or the file is a VSAM file
REMOTENAME	CHAR(8)	The name by which the file is known to the CICS region that is named in the REMOTESYSTEM field
REMOTESYSTEM	CHAR(4)	The name of the CICS region in which the file is defined
REMOTETABLE	CHAR(10)	Indicates whether the file represents an open remote data table. The value is REMTABLE (an open remote data table)
RLSACCESS	CHAR(10)	Indicates whether the file is defined to be opened in RLS mode. The values are: NOTAPPLIC, NOTRLS, or RLS
STRINGS	INTEGER	Indicates the number of strings (concurrent operations) that are specified for the file

Column	Type	Description
TABLE	CHAR(10)	Identifies the type of table set that corresponds to this file. The values are: CFTABLE, CICSTABLE, NOTTABLE, or USERTABLE
TABlename	CHAR(8)	The name that is specified for the coupling facility data table
TYPE	CHAR(10)	The type of data set that corresponds to this file. The values are: ESDS, KEYED, KSDS, NOTKEYED, RRDS, VRRDS, or NOTAPPLIC
FIRST_RUN	TIMESTMP	Time of first occurrence
LAST_RUN	TIMESTMP	Time of latest observation

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples:

- Show the transactions that access dataset CICSTLS.CICSTS0A.TTZMSGs, how they access this dataset, and the file name:

```
SELECT DISTINCT TRANSID,
        PROGRAM,
        FILE_NAME,
        FUNCTION,
        DSNAME
FROM CIU_CICS_DATA A, CIU_FILE_DETAIL B
WHERE FILE_NAME = 'TTZMSGs' AND
      DSNAME = 'CICSTLS.CICSTS0A.TTZMSGs' AND
      A.OBJECT = FILE_NAME;
```

- Show all files not used since a given date:

```
SELECT DISTINCT B.FILE_NAME,
        B.DSNAME,
        DATE(A.LAST_RUN)
FROM CICSIA.D.CIU_CICS_DATA A,
     CICSIA.D.CIU_FILE_DETAIL B
WHERE A.TYPE = 'FILE' AND
      DATE(A.LAST_RUN) < '2008-04-01' AND
      A.OBJECT = B.FILE_NAME;
```

7.4 Scanner base tables

In this section, we look at tables that the Scanner component of CICS IA creates.

We only cover a few of the tables in this section. For details about all of the Scanner base tables, see the *CICS Interdependency Analyzer for z/OS User's Guide and Reference*, SC34-6790-01.

7.4.1 CIU_SCAN_SUMMARY

Table 7-13 contains summary information about every scanned load module.

Table 7-13 The CIU_SCAN_SUMMARY table

Column	Type	Description
DSNAME	CHAR(44)	The data set name
PROGRAM	CHAR(8)	Module name
LANGUAFE	CHAR(10)	Programming language detected
LE	CHAR(7)	Language environment (LE) detected
CICS_OR_BATCH	CHAR(5)	CICS transaction or batch
AFFINITY_COUNT	INTEGER	Number of commands with potential to create affinities
MVS_POST_COUNT	INTEGER	Number of MVS POST commands
DEPENDENCY_COUNT	INTEGER	Number of commands with potential to create dependencies

Sample queries

In this section, we show details of the sample queries that are provided with CICS IA. The two samples:

- ▶ Show all scanned programs that are written in PL/I:

```
SELECT PROGRAM,  
       DSNAME  
FROM CIU_SCAN_SUMMARY READONLY  
WHERE LANGUAGE = 'PL/I';
```

- ▶ Show all programs that were scanned but not used so far (not picked up by the collector):

```
SELECT PROGRAM
```

```
FROM CIU_SCAN_SUMMARY READONLY
WHERE PROGRAM NOT IN
  (SELECT PROGRAM
   FROM CIU_CICS_DATA READONLY);
```




CICS IA usage scenarios

In this chapter, we provide an overview of the ways that you can use the CICS Interdependency Analyzer for a variety of scenarios. This chapter contains the following sections:

- ▶ Thread safety analysis
- ▶ CICS Web services assistance
- ▶ Migration assistance

8.1 Threadsafety analysis

CICS provides the Open Transaction Environment (OTE), which is an architecture that can be exploited by certain applications, TRUEs and CICS system programs. OTE was introduced for three main purposes:

- ▶ Increasing transaction throughput by enabling more concurrency
- ▶ Introducing the possibility to use non-CICS APIs within a CICS system
- ▶ Improving the performance of CICS applications and the system as a whole

Programs that exploit OTE have to be *threadsafe*, meaning that they can execute their own logic under an *open* TCB that OTE manages. In addition to this, CICS also provides a range of its EXEC CICS API and SPI that is threadsafe, and so can be executed under an open TCB that is associated with a threadsafe program or under the QR TCB for quasi-rent applications. Some programs are suitable for OTE-exploitation, and there are other programs that are not suitable. You can use CICS IA to help identify those programs that make suitable candidates to exploit OTE.

8.1.1 Background to the CICS Open Transaction Environment

Prior to OTE, all CICS application code ran under the main CICS TCB called the quasi-reentrant (QR) TCB, apart from some specific VSAM execution and other specialized CICS activity. The CICS dispatcher automatically sub-dispatched the QR TCB between the different CICS tasks. It performed this multi-tasking work extremely rapidly, giving the appearance of many tasks executing at once under CICS. In reality, only one task was executing on the QR TCB at any point-in-time.

Being a cooperative programming model, each task voluntarily relinquished control when it issued a CICS service. This might cause a CICS dispatcher to wait and hence the opportunity for another task to be dispatched under the QR TCB.

By definition, the one QR TCB in a given CICS system could execute on only one Central Processor (CP) at a time; therefore, CICS execution was unable to exploit multiple physical CPs for the majority of its processing. In addition, CICS had to ensure that the QR TCB did not have to perform *blocking* work that stopped other applications from executing. And yet, there was a need to perform work that would block a TCB, for example, when invoking SQL calls to perform DB2 requests from within a CICS application, which meant that a group of subtask TCBs had to be managed by the CICS/DB2 attachment facility to transfer the thread of execution off of the QR TCB for the duration of a DB2 request.

Note: Blocking means a TCB is halted by a MVS wait.

Quasi-reentrant programs always run under the QR TCB and can access shared resources, such as the Common Work Area (CWA) or shared storage obtained through EXEC CICS GETMAIN SHARED, safe in the knowledge that they are the only quasi-reentrant program that is running at that point-in-time because using the QR TCB guarantees serialized access to those shared resources. An example of this concept is a program that updates a counter in the CWA. The program can reference or update this counter without fear of interruption by another task, and when it gets suspended by the CICS dispatcher, it knows the counter still has the value that was assigned by it. However, the trade off to such inherent serialization is the limitation of the one TCB under which to sub-dispatch the CICS workload.

To relieve the QR TCB and to improve the management of subtask TCBs, such as those that are required by the CICS/DB2 attachment facility, OTE introduced a new class of TCB that applications and TRUEs, such as DB2, can use. These are called *open TCBs*. An open TCB is assigned to a CICS task for its sole use. Multiple open TCBs can run concurrently in CICS. There are several modes of open TCBs that are used to support various functions, such as JVMs in CICS, open API TRUEs and programs, and C and C++ programs that were compiled with the XPLink option. Unlike the QR TCB, there is no sub-dispatching of other CICS tasks on an open TCB.

Each open TCB can execute on a CP in parallel with work on other CPs within the system, which lets CICS exploit true parallel processing. True parallel processing is unlike the multi-tasking programming model that the QR TCB uses, where only one task is actually executing on the CP at any one time. This gives the potential of increased throughput for a single CICS system, as long as the necessary computing power is available.

OTE in CICS is an ongoing implementation, staged over several releases of CICS Transaction Server:

- ▶ Stage 1 - OTE function introduced for Java. Some EXEC CICS commands were made threadsafe. Delivered in CICS TS 1.3
- ▶ Stage 2 - OPENAPI TRUEs could exploit OTE. Further EXEC CICS commands were made threadsafe. Delivered in CICS TS 2.2 and 2.3
- ▶ Stage 3 - Full application use of open TCBs. Further threadsafe EXEC CICS commands. Delivered in CICS TS 3.1

A *threadsafe application* can be defined as a collection of programs that employ an agreed-upon form of serialized access to shared application resources. A program written to *threadsafe standards*, then, is a program that implements these serialization techniques. It is important to understand that a single program that operates without the agreed-upon serialization technique(s) can affect the predictability and therefore integrity of an entire system of otherwise threadsafe programs. Therefore, an application system cannot be considered *threadsafe* until all programs that are sharing common resources implement appropriate standards.

Important rule: Only after it is fully understood whether an application is threadsafe, and all access to all shared resources are serialized, should any of the application's programs be defined as threadsafe. Failure to follow this rule might result in unpredictable results and put the integrity of application data at risk.

CICS assumes responsibility for ensuring the integrity of all of the resources that it itself manages. Therefore, for threadsafe EXEC CICS commands, CICS code is amended to ensure that its own appropriate serialization takes place. For non-threadsafe EXEC CICS commands, CICS switches execution to the serialized QR TCB to perform the command.

Using non-threadsafe EXEC CICS commands can have a performance penalty, depending upon the application because CICS needs to switch back to the QR TCB when a non-threadsafe CICS command is executed. If there are many non-threadsafe CICS commands in a program that is otherwise threadsafe, the extra switching back to the QR TCB has a detrimental effect on performance. However, because CICS serializes its own logic and resources where appropriate, there is no risk to data integrity when using non-threadsafe EXEC CICS commands from within threadsafe applications.

When a threadsafe application executes on an open TCB and invokes threadsafe EXEC CICS commands, the result is a reduced number of TCB switches between the open TCB and QR TCB. This gives a reduction in CPU consumption that corresponds to the number of saved TCB switches. The more CICS commands that are made threadsafe and exploited by threadsafe applications, the greater the probability that the program remains executing on an open TCB.

Reducing the CPU consumption of an application does not necessarily result in improved response times. An application can be a heavy user of CPU, but if the processor has spare capacity and the application is not CPU constrained, then a reduction in path length might have a negligible impact on response times. For many, however, the financial cost that is incurred in running their applications is

related to the amount of CPU they consume. In these circumstances, the CPU savings that are gained by migrating appropriate applications to a threadsafe environment can indeed equate to a financial saving.

Those who might benefit most from migrating to a threadsafe environment are those who experience poor response times for any of the following reasons:

- ▶ The QR TCB is CPU constrained, which means that the QR TCB is consistently waiting to be dispatched by z/OS. Every task that runs under the QR TCB is being delayed.
- ▶ Applications are waiting excessively for the QR TCB. Although the QR TCB is not itself constrained by CPU, applications are contending for their share of QR use.
- ▶ The CICS region, in general, is CPU constrained. The system as a whole is approaching 100% busy, and CICS is being constrained along with everything else.

Prior to OTE, one way to avoid QR TCB contention was to provide additional application owning regions (AORs) and use workload balancing techniques to spread work across them. With OTE, the ability to define transactions as threadsafe and process as many tasks as possible on open TCBs helps to remove the constraint on the QR TCB for a given CICS region, which can reduce the response times of both threadsafe and non-threadsafe transactions as a result.

Depending on how an application is designed, defining it as threadsafe can significantly reduce the path length of execution with corresponding improvements in throughput, performance, and response times. IBM recommends that all new application programs are written to threadsafe standards at whatever level of CICS they are developed.

8.1.2 Using CICS IA to assist with threadsafety analysis

There are many reasons why you might want to analyze your program inventories to understand their characteristics. One might be to review which EXEC CICS commands are invoked by your applications, as part of a wider piece of development work to make particular applications threadsafe, with the goal of redefining them as such to CICS and so allowing them to exploit OTE. CICS IA provides support to help identify those aspects of application programs that relate to the use of OTE.

A goal of OTE exploitation is to improve application performance, which might be quantified by a reduction in CPU usage, increased transaction throughput, or better response times for the users of a program. By redefining an application to

be threadsafe, these goals can potentially be achieved. However, this depends upon the nature of the application, and in particular the commands that it invokes (both EXEC CICS, EXEC SQL, and calls to WMQ). The nature of these commands (and the order in which they are issued) has a direct influence on the benefits that might be seen if the application were to be redefined as threadsafe.

Important reminder: Redefining an application to be threadsafe should only be carried out after the application logic is properly studied (and changed if necessary) to ensure that it is indeed suitably serialized when handling shared resources. Analyzing which commands the program issues is important but not the whole story.

The CICS command level API and SPI are very rich and contain many EXEC CICS commands, which are documented in the *CICS Transaction Server for z/OS Application Programming Reference*, SC34-6819 and *CICS Transaction Server for z/OS System Programming Reference*, SC34-6820, for CICS Transaction Server V3. The API command syntax diagrams contain the statement This command is threadsafe if the command is a threadsafe one. The *System Programming Reference* lists the threadsafe EXEC CICS SPI commands. Issuing such threadsafe commands when running under an open TCB does not result in a TCB switch back to QR in order for CICS to process them, which allows for better exploitation of open TCBs.

With a high percentage of such threadsafe commands within a threadsafe application, the thread of execution can remain on an open TCB for longer and so improve the performance of the system. When you review which applications might be suitable candidates for threadsafety, an important part of the analysis work is to identify which commands the applications exploit.

In a similar way, to DFHEISUP and the use of filter table DFHEIDTH, CICS IA can detect CICS command use that are threadsafe inhibitors, which are those commands that *might* cause a program not to be threadsafe because they allow access to shared storage.

While CICS IA has an offline scanner that performs a similar role to DFHEISUP, another advantage CICS IA brings is the runtime collector component. The runtime collector component reports on executed applications rather than applications that reside in load libraries, some of which might be obsolete.

CICS IA reporting provides two means to identify which EXEC CICS commands are threadsafe (or not) within an application:

- ▶ Query the V_CIU_SCAN_TRDSAFE view.
- ▶ Run the 'Threadsafe Dynamic Summary/Detail' report against the CIU_PROGRAM_DETAIL table and CIU_CICS_DATA table

Querying the V_CIU_SCAN_TRDSAFE view takes data from the join of the CIU_SCAN_DETAIL table and the CIU_THREADSafe_CMD table. This is data that the Load Module Scanner (LMS) produced, which runs as a batch job and scans a target program library. The Scanner looks for the bit patterns that match the argument_zero (arg0) bitstrings that define particular EXEC CICS commands. As such, it is occasionally possible for the Scanner to report on the presence of such commands when in fact it scanned program data that just happens to have matching bitstrings to the arg0 values of these EXEC CICS commands. (Such false-positive results are a rarity, but they can possibly be seen in some cases).

The various types of commands that might be reported on are as follows:

- ▶ 'Threadsafe' calls are EXEC CICS commands that do not cause a TCB switch from an open TCB back to the QR TCB in order for CICS to execute them.
- ▶ 'Non-threadsafe' calls are EXEC CICS commands that do cause a switch back to the QR TCB if issued under an open TCB.
- ▶ 'Indeterminate threadsafe' calls are EXEC CICS commands, where it cannot be determined if they would result in a TCB switch back to QR or not, which is because there is insufficient information that is available for CICS IA to make such a decision, for example, the Scanner cannot tell whether an EXEC CICS WRITEQ command is directed towards a temporary storage queue that will be dynamically defined as remote. The queue location might be dynamically established at runtime. Because part of the CICS function-shipping code path is itself non-threadsafe, potentially function-shippable threadsafe commands are therefore indeterminable by the Scanner.
- ▶ 'Threadsafe Inhibitor' calls are EXEC CICS commands that indicate that an application might perform work that prevents it from being redefined as threadsafe. These commands are EXEC CICS ADDRESS CWA, EXTRACT EXIT, GETMAIN SHARED, and LOAD. Further analysis at the application level is required, particularly to determine the presence (or absence) of suitable serialization techniques in relation to the use of these commands.

Sample queries are provided in CIUSAMPC to detect non-threadsafe and indeterminate threadsafe commands.

Figure 8-1 on page 190 shows part of the results from a sample query against the V_CIU_SCAN_TRDSAFE view. In this example, the indeterminate threadsafe commands have been enquired upon. The target programs can be seen to contain the command signatures for particular EXEC CICS commands that when executed might or might not be treated as threadsafe by CICS. In the case of an EXEC CICS LINK command, this is threadsafe if the program being linked to resides on the same CICS system and is defined to be threadsafe. In other words, the thread of execution for a LINK command can remain on an open TCB during the execution of a LINK to a local threadsafe application program. If

the EXEC CICS LINK is to a program on a remote CICS system however, the command is not threadsafe because it has to be function shipped there. Whether the remote program being linked to is defined as quasirent or threadsafe is then irrelevant - the act of function shipping a command level request requires CICS to execute part of its internal logic that is not threadsafe, so it must switch to the QR TCB to serialize its processing.

```
-- q3a- Show all programs with indeterminate-threadsafe EXEC CICS
-- commands
SELECT 'TS32 INDETERMINATE-THREADSAFE CALLS FOR ',
      "PROGRAM", OFFSET, COMMAND, ' = ', COUNT(*)
FROM V_CIU_SCAN_TRDSAFE READONLY
WHERE CICS_TS32 = 'I'
AND DSNAME='CICSIAD.MANA.LOAD'
GROUP BY "PROGRAM", OFFSET, COMMAND;
```

	PROGRAM	OFFSET	COMMAND		
TS32 INDETERMINATE-THREADSAFE CALLS FOR	LGACUS01	640	WRITE	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	LGAPOL01	814	WRITE	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	LGDPOL01	610	DELETE	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	LGICUS01	630	READ	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTC1	845	LINK	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTC1	862	LINK	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTP1	1056	LINK	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTP1	1073	LINK	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTP1	1090	LINK	=	1
TS32 INDETERMINATE-THREADSAFE CALLS FOR	SSTESTP1	1107	LINK	=	1

Figure 8-1 Results from sample query against the V_CIU_SCAN_TRDSAFE view

The results in Figure 8-1 are from data that the Scanner extracted, analyzing the static bit pattern content within the target programs in a library. CICS IA also performs dynamic analysis at CICS runtime and gathers data that can be post-processed with the Threadsafe Dynamic Analysis report. The report provides options for specifying PROGRAMNAME and REGIONNAME, both of which support wildcard characters (* is the default in both cases, meaning that all are to be returned). The report also allows the CICSLEVEL to be selected (3.1 or 3.2), and provides the user with the ability to select whether to produce a SUMMARY or DETAIL report, and the appropriate value for LINESPERPAGE (the default is 66).

Figure 8-2 on page 191 shows an extract of some of the results when specifying the SUMMARY option for the report. The summarized results for programs SSTESTC1, SSTESTP1, and SSTESTP2 are shown. These programs executed on CICS applid IYDZZ328. They each issued various EXEC CICS commands, which were then analyzed as being either threadsafe, non-threadsafe, or indeterminate threadsafe commands. The report also shows the number of calls to DB2, WMQ, and IMS that these programs made. Figure 8-2 on page 191

displays the number of dynamic calls that a program made (a natural language CALL from a COBOL program to another program in a separate load module is an example of such a dynamic call). Finally, the number of threadsafe inhibitor calls made (such as EXEC CICS ADDRESS CWA) are listed too.

This information helps reveal the runtime nature of each program and assists in determining the suitability of programs that are being redefined as threadsafe in CICS.

CICS INTERDEPENDENCY ANALYZER VERSION 2.2.0							2008/04/22:09.34.32		PAGE 2	
Program Dynamic Analysis - THREADSAFE SUMMARY LISTING FOR CICS TS 3.2										
APPLID	Program	Linkedit Date	Execution Key	Concurrency	APIST	Storage Protect	CICS Rel	LIB	Dataset Name	
IYDZZ328	SSTESTC1	-----	USER	QUASIRENT	CICSAPI	ACTIVE	0650	CICSIAD.MANA.LOAD		
	Total CICS calls:		12	Threadsafe:	1	Non-Threadsafe:		9	Indeterminate Threadsafe:	2
				DB2 calls:	0	MQ calls:		0	IMS calls:	0
				Dynamic Calls:	0	Threadsafe Inhibitor calls:		0		
IYDZZ328	SSTESTP1	-----	USER	QUASIRENT	CICSAPI	ACTIVE	0650	CICSIAD.MANA.LOAD		
	Total CICS calls:		29	Threadsafe:	1	Non-Threadsafe:		23	Indeterminate Threadsafe:	5
				DB2 calls:	0	MQ calls:		0	IMS calls:	0
				Dynamic Calls:	0	Threadsafe Inhibitor calls:		0		
IYDZZ328	SSTESTP2	-----	USER	QUASIRENT	CICSAPI	ACTIVE	0650	CICSIAD.MANA.LOAD		
	Total CICS calls		23	Threadsafe:	1	Non-Threadsafe:		17	Indeterminate Threadsafe:	5
				DB2 calls:	0	MQ calls:		0	IMS calls:	0
				Dynamic Calls:	0	Threadsafe Inhibitor calls:		0		

Figure 8-2 Some extracted results from the Threadsafe Dynamic Analysis report - Summary

Figure 8-3 on page 192 shows an extract of some of the results for program SSTESTC1 when specifying the DETAIL option for the report. The detailed report breaks down each of the calls that the program makes, which gives the associated resource names, the offsets of the commands, the program attributes (length and use count), and the threadsafe characteristics of each of the calls.

The detailed report option is useful when attention is to be paid to a particular program of interest, having analyzed the system as a whole with the summarized version of the report.

Using CICS IA data in this way, the static content (potential commands) and runtime behavior (executed commands and calls) can be studied to help construct a picture of the applications under review coupled with a detailed analysis of the business logic within the application programs. Remember that the analysis of what commands that the applications use is only part of the story when considering whether a program is suitable for being redefined to exploit threadsafety within CICS. If a program contains threadsafe commands, and looks like a good candidate to exploit OTE based upon its runtime

characteristics, you must still carefully analyze it to reveal evidence of suitable serialization techniques when handling its state data and shared resources.

Important point to remember: *There is no way to safely avoid this step.*

```

CICS INTERDEPENDENCY ANALYZER VERSION 2.2.0                                2008/04/22:09.35.44    PAGE
3
Program Dynamic Analysis - THREADSAFE DETAIL LISTING FOR CICS TS 3.2

APPLID  Program  Linkedit  Execution  Concurrency  APIST  Storage  CICS  LIB Dataset Name
         Date      Key                               Protect  Rel

-----
Threadsafed  CMD  Function              Type      Resource              Offset  Program  Use
                                     Length  Count

-----
IYDZZ328 SSTE1C1 ----- USER      QUASIRENT  CICSAPI ACTIVE  0650  CICSIA.D.MANA.LOAD
CICS LINK          PROGRAM  LGACUS01          A34   1540    1  I
CICS LINK          PROGRAM  LGICUS01          90A   1540    1  I
CICS RECEIVE       MAP      SSMAPC1           87E   1540    1  N
CICS RECV MAP      MAPSET   SSMAP             87E   1540    1  N
CICS RETURN        TRANSID  SSC1              BD0   1540    1  Y
CICS SEND          MAP      SSMAPC1           AA0   1540    1  N
CICS SEND          MAP      SSMAPC1           744   1540    1  N
CICS SEND          MAP      SSMAPC1           984   1540    1  N
CICS SEND          TEXT     SEND TEXT         C24   1540    1  N
CICS SEND MAP      MAPSET   SSMAP             AA0   1540    1  N
CICS SEND MAP      MAPSET   SSMAP             744   1540    1  N
CICS SEND MAP      MAPSET   SSMAP             984   1540    1  N
Total CICS calls:  12  Threadsafed:  1  Non-Threadsafed:  9  Indeterminate Threadsafed:  2
                   DB2 calls:  0  MQ calls:  0  IMS calls:  0
                   Dynamic Calls:  0  Threadsafed Inhibitor calls:  0

```

Figure 8-3 Some extracted results from the Threadsafed Dynamic Analysis report - Detail

It is worth summarizing the threadsafed status of the various interfaces to other subsystems that are returned by the report. The CICS/DB2 attachment, CICS/IMS DBCTL interface, and CICS/WMQ adapter all implement Task Related User Exits to allow applications to invoke calls to the other subsystems. Depending upon the release of CICS that is being used, the status of the various TRUEs vary from quasirent to threadsafed, for example, the CICS/DB2 adaptor was restructured to be an OPENAPI-enabled TRUE in CICS TS 2.2. Table 8-1 on page 193 shows the TRUE threadsafed status for the various CICS releases.

Table 8-1 Summary of the threadsafe status for the CICS/DB2, IMS and WMQ interfaces

	CICS TS 2.2	CICS TS 2.3	CICS TS 3.1	CICS TS 3.2
IMS	N	N	N	N
DB2	Y	Y	Y	Y
WMQ	N	N	N	Y

8.1.3 Using the CICS IA Explorer to assist with threadsafety analysis

You can use the CICS IA Explorer to assist with the analysis of threadsafety. You can use it to display details of programs that might have threadsafe data integrity issues, for example, those programs that issue commands, such as EXEC CICS ADDRESS CWA, which indicate the need to consider the use of appropriate serialization techniques around logic that affects shared data areas and storage.

Figure 8-4 on page 194 shows an example CICS IA Explorer window that details those programs that might have such concerns. In this example, the query is against all such commands, and the window shows the resource types that are associated with each command instance from every such program that was detected.

The CICS IA Explorer also provides the ability to review commands issued by TCB mode and program. With OTE, the TCB mode is important because the goal of OTE exploitation is to utilize open TCBs, where possible, and to help free up the QR TCB to process quasi-reentrant workloads that are not threadsafe and so must run in a serialized manner under protection of the QR TCB.

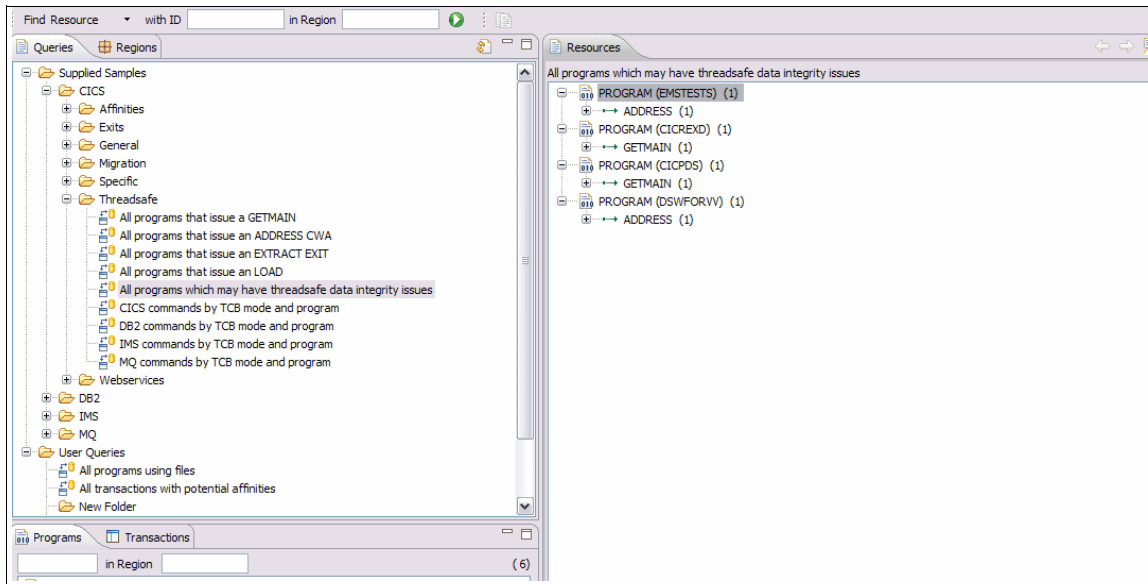


Figure 8-4 An example of the use of the CICS IA Explorer for threadsafety analysis

8.2 CICS Web services assistance

CICS IA can assist with the analysis of Web services support that was introduced in CICS Transaction Server Version 3 (CICS TS).

CICS TS can act as both a Web services requester, when outbound requests to Web services providers are made from within CICS, and also as a Web services provider, when CICS acts as the server for inbound Web services requests to it. CICS IA provides support for both of these variants of Web services.

There are three resource types that relate to Web services support in CICS TS: the WEBSERVICE, PIPELINE, and URIMAP resources types.

A WEBSERVICE resource defines the execution environment to allow a CICS application program the ability to act as a Web services provider or requester. The Web service interaction is formally described with Web service description language (WSDL).

A PIPELINE resource definition provides information about the processing that will act on a Web service request and on the response. It is used for both Web services requester and provider operations. A typical PIPELINE definition describes an infrastructure that many applications can utilize.

A URIMAP definition matches a Uniform Resource Identifier (URI) to a WEBSERVICE definition and a PIPELINE definition. There should be a unique URI for each Web service to be used in CICS.

Together, these CICS resource types provide the internal support for the implementation of CICS Web services. In turn, CICS IA provides its own Resource Option command type of 'WEB Services', to assist with the analysis of Web services support in CICS TS.

Using the CINT transaction, you can set this option to N (for No), Y (for Yes), or D (for Yes plus Detail). As with the other Resource Options, leaving the command type option field blank causes the default value to be used instead. Figure 8-5 on page 196 shows an example of the CINT Resource Options window, which contains the command type fields for the various options that CICS IA can detect. In this example, the 'WEB Services' Resource Option value is set to D.

If the 'WEB Services' Resource Option setting is specified, CICS IA can provide assistance in the analysis of and preparation for the use of Web services within CICS, which is carried out in a variety of ways.

```

CIU240          CICS Interdependency Analyzer for z/OS - V2R2M0  2008/05/05
                  CICS Resources Options for                    08:27:09AM
                  CICS Sysid   : Z138   CICS Applid   : ANDYCICS

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

Detect command types: Y=Yes, N=No or blank=default
                      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 . . . . . N *WEB Services . D
Exits . . . . . Y Others . . . . . Y

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 . . . . . N Journals . . . Y
Library . . . Y IPCONN . . . . Y

CICS Sysid: Z138 CICS Applid: ANDYCICS TermID: V109

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

```

Figure 8-5 Example of the CINT Resources Options window showing the WEB Services command option

8.2.1 Web services requester detection

CICS IA can assist with the detection of Web services requests that are made from within CICS TS. The EXEC CICS API, which is provided to perform such Web service requests, is the EXEC CICS INVOKE WEBSERVICE command. The command specifies the name of a WEBSERVICE resource, which contains information about the service to be invoked.

CICS IA detects Web services requests when an application issues an EXEC CICS INVOKE WEBSERVICE command and the 'WEB Services' Resource Option is set to Y or D. CICS IA captures the information about the usage of the command and saves the data into the CIU_CICS_DATA table.

CICS IA captures additional information about Web services resources and places it in the CIU_WEBSERV_DETAIL table. As with the other primary resource types tables, this is a repository for information that is collected through inquiries against the resource type, and the data is collected at the time the resource is referenced.

Note: Such detailed information is only collected when the appropriate CICS Resource Options value is set to D.

8.2.2 Web services provider detection

Unlike the EXEC CICS INVOKE WEBSERVICE command, there is no equivalent CICS API command to represent the provision of support for an incoming Web services request. The CICS Web services support within CICS TS processes and handles this work, in particular by the Pipeline domain.

CICS TS 3.2 was extended through APAR PK56097 / PTF UK33060, which provides support for a new CICS Global User Exit point XWSPRROO, together with sample exit DFH£PIEX, to assist with monitoring CICS Web services activity.

The CICS Pipeline domain exit can be used to customize the processing that occurs for Web services within the pipeline. The XWSPRROO exit enables you to access containers on the current channel after the Web services provider application issues the Web service response message and before CICS creates the body of the response message.

The exit can be used to issue EXEC CICS API and SPI commands to examine the containers on the current channel. CICS IA enables this exit for its analysis of Web services provider activity within CICS TS 3.2. The exit captures the equivalent data to that recorded for CICS Web services requester commands and saves the data into the CIU_CICS_DATA table.

8.2.3 Identifying candidate Web services programs

Some CICS application programs make better candidates for Web services than do others. Typically, the ones that are more suitable are those programs that contain some particular form of business logic implementation, but not any of the corresponding presentation logic to communicate the results. In this way, they adhere to the sensible programming architectural model of separating business and presentation logic into different application programs.

This approach was advocated before the introduction of Web services support within CICS TS. It allows for much better flexibility when you implement your applications and assists with workload balancing and other system management requirements. If the presentation and business application layers are connected by an EXEC CICS LINK command, this provides many benefits. These include workload balancing using DPL, XEIN, and XEIOU GLUE points, data sharing

using a commarea or channel, CICS trace diagnostics, provision for EDF debugging support, and so on.

CICS IA provides sample query data to assist in identifying those applications that might make good candidates to be deployed as Web services providers. Such applications have characteristics, such as not containing EXEC CICS commands, that indicate the existence of associated presentation logic. They are also programs that are passed through a commarea or channel when they are invoked.

Example 8-1 shows the sample SQL query CIUSAMP0 that is provided in SCIUSQL.

Example 8-1 SQL query CIUSAMP0 is provided in SCIUSQL

```
SELECT * FROM CIU_CICS_DATA
WHERE (COMMAREA = 'Y' OR CHANNEL = 'Y')
AND OBJECT NOT IN
  (SELECT "PROGRAM" FROM CIU_CICS_DATA
   WHERE
     (FUNCTION IN ('SEND' ) AND
      TYPE IN ('MAP',
              'TEXT' ) )
    OR (FUNCTION IN ('RECEIVE' ) AND
      TYPE IN ('MAP',
              ' ' ) )
    OR (FUNCTION IN ('ADDRESS' ) AND
      OBJECT IN ('TCTUA' ) )
    OR (FUNCTION IN ('ASSIGN' ) AND
      OBJECT IN ('ALTSCRNHT',
                'ALTSCRNWD',
                'DEFSCRNHT',
                'DEFSCRNWD',
                'APLKYBD',
                'APLTEXT' ) )
  );
```

The SQL construct looks for the presence of a channel or commarea that is associated with the program and the absence of a command, such as an EXEC CICS SEND MAP, EXEC CICS RECEIVE MAP, EXEC CICS ADDRESS TCTUA, and so on. These are some of the commands that give an indication of associated presentation logic within an application.

The *CICS Application Programming Reference* contains a description of all of the CICS API commands. You can use this reference to help you to understand what commands your applications use and to review the scope of any presentation logic that is contained within them.

8.3 Migration assistance

CICS IA can assist with the migration from one release or version of CICS to another. It helps select those applications that require more or specific testing during the migration process by focusing on relevant commands and exits that that the CICS system uses.

8.3.1 Migration drivers and inhibitors

You have many different reasons for wanting to migrate the levels of your CICS environments. You might want to exploit new functionality, either in the associated hardware and operating system or by using newly developed functions that are provided by enhancements to the EXEC CICS or EXEC CPSM APIs. You might use third-party software packages that require a higher level of CICS in order to operate, or you might need to migrate in order to remain at an officially supported level of the product or to meet changing Service Level Agreements (SLAs). Reasons for migration can therefore include business, technological, financial, and legal factors. There is no single driving force for every user. Each migration scenario is different, in terms of its cause, requirements, scope, time frame, and budget.

Depending upon the size and complexity of installations, the process of migrating all CICS and CPSM environments can take anywhere from several months upwards to multiple years. The migration process includes preparation and planning, education, testing, implementing, verifying, and finally deploying into production.

For large sites, after this process is completed for a given release, the passing of time can mean that preparation to repeat the process for the next migration of the product is necessary. This continuous migration cycle can cause some to skip releases within the product, simply because there is insufficient resources or time available to migrate as frequently as new releases become available. In large or complex environments, this can lead to users remaining on lower levels of CICS and CPSM for longer, which results in the additional planning exercise of having to migrate across multiple release levels (or version levels) of the product.

It is important to minimize the risk that a migration can lead to unexpected problems, such as incompatibility in data formats or file structures or an unacceptable performance degradation within the code path. CICS IA can assist in the preparation work that is a key part in any migration planning and help mitigate this risk and improve the migration process.

8.3.2 CICS IA application migration support

Most of you want to know where to focus your regression testing efforts on the areas of a new product release or version that changed the most, which potentially might have an effect upon migrating. You also want to know which of your applications might potentially be affected by associated changes to the CICS API and SPI. Although the intention is to maintain compatibility for existing programs, there are some occasions when the official interfaces into CICS have to change, for example, RESP2 values can be added, commands and options can become obsolete, and so on. In extreme cases, interfaces can be removed completely, which was the case with the original macro level programming API after the EXEC CICS command level API became firmly established as the strategic and extendable means of writing CICS applications.

Although such changes are well documented and publicized by IBM, it can be a major effort to analyze large numbers of applications and review their usage of the API. The different sets of changes that you can encounter depending upon which level of CICS you are migrating from and which level you are migrating to can also complicate things.

CICS IA can assist with this analysis. It provides sample queries for the different migration combinations that can be made. The queries help to identify those programs that make use of EXEC CICS API and SPI options that changed between the two particular levels of CICS.

These sample SQL queries are provided for the following combinations of migration strategies:

- ▶ CIUM1331 - migration from CICS TS 1.3 to CICS TS 3.1
- ▶ CIUM1332 - migration from CICS TS 1.3 to CICS TS 3.2
- ▶ CIUM2231 - migration from CICS TS 2.2 to CICS TS 3.1
- ▶ CIUM2232 - migration from CICS TS 2.2 to CICS TS 3.2
- ▶ CIUM2331 - migration from CICS TS 2.3 to CICS TS 3.1
- ▶ CIUM2332 - migration from CICS TS 2.3 to CICS TS 3.2
- ▶ CIUM3132 - migration from CICS TS 3.1 to CICS TS 3.2

The sample SQL queries assist in identifying obsolete commands and options (such as terminal control commands for BTAM devices), changes to commands (such as file control requests for extended ESDS files using the new XRBA option), and changed or obsolete SPI commands and options, such as for files, document templates, programs, and so on.

In this way, the focus for research and analysis work, as part of the application migration effort, can be applied towards those specific applications whose constructions are identified by CICS IA. This in turn can assist in helping to plan the appropriate areas in which to concentrate any regression testing work.

One approach to using CICS IA as part of the migration assistance is to:

1. Collect the appropriate data for the applications used in the CICS system that is to be migrated.
2. Run the CICS IA SQL queries to identify those applications that might need changing or at least further investigating.
3. Evaluate the applications, and if necessary change them.
4. Run the appropriate unit testing for any such changes, and then integrate the changes back into the system and rerun the appropriate system tests.
5. Migrate the CICS system.
6. Run the regression test cases for the applications of interest as part of the overall testing of the new CICS environment.

8.3.3 Identifying TRUEs and GLUEs

CICS IA can also assist detecting those exits that are used within CICS, both Task Related User Exits (TRUEs) and Global User Exits (GLUEs).

There are various types of events when CICS IA collects data that relates to the use of exits within CICS, which are:

- ▶ At the time that CICS IA is started. When this occurs, CICS IA can gather data on those exits that were already started prior to this point.
- ▶ When an EXEC CICS ENABLE command is issued after CICS IA is started.
- ▶ When an application calls a TRUE.

With these various collection points, CICS IA can gather specific data on the use of exits within the CICS system, which can assist as part of the migration planning determination of what exits are in use, their prevalence, scope, and so on.

CICS IA stores its captured exit data in the CIU_EXIT_INFO table in DB2.

Note: The Exits command type on the CICS Resources Options panel must be set to Y to allow the detection of exit activity by CICS IA.



Affinities

One of the functions of CICS IA is the ability to analyze affinity data. The affinity data is loaded into DB2 databases, which you can perform SQL queries on and produce detailed reports from.

In this chapter, we briefly explain what affinities are and show how to create and install an input file for CICSplex SM with the affinity-transaction-group definition.

9.1 Overview of transaction affinities

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

There are two categories of affinity:

- ▶ *Inter-transaction affinity*
- ▶ *Transaction-system affinity*

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 that there is no restriction in the choice of available target regions. However, even when transaction affinities do exist, there are limits to the scope of these affinities determined by the:

- ▶ Affinity relations
- ▶ Affinity lifetime

CICS IA cannot detect affinities in the following types of dynamically routed requests:

- ▶ Non-terminal-related START requests
- ▶ Distributed program link (DPL) requests
- ▶ Method requests for enterprise beans or CORBA stateless objects

For these types of dynamically routed requests, review your application to determine whether it is suitable for dynamic routing.

9.1.1 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 affinities, which impose restrictions on the dynamic routing of transactions, can occur in the following circumstances:

- ▶ One transaction terminates and leaves “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 to work safely, the first transaction usually waits on an event, which the second transaction posts when it has read the data that the first transaction created. This synchronization technique requires that both transactions are routed to the same CICS region.

9.1.2 Transaction-system affinity

A transaction-system affinity is an affinity between a transaction and a particular CICS region. It is caused by the transaction interrogating or changing the properties of the CICS region.

Transactions with an affinity to a particular CICS region, rather than to another transaction, are not eligible for dynamic transaction routing. Typically, they are transactions that use CICS SPI commands, such as EXEC CICS INQUIRE or SET, or that depend on global user exit programs.

9.1.3 Affinity relations

When a transaction is associated with an affinity, the affinity relation determines how the dynamic routing program selects a target region for an instance of the transaction. An affinity relation can be classified as one of the following:

Global	A group of transactions in which all instances of all of the transactions in the group that are initiated from any terminal, or are BTS or Link3270 transactions, must execute in the same target region for the lifetime of the affinity. The affinity lifetime for global relations can be “system” or “permanent.”
BAPPL	All instances of all transactions in the group are associated with the same CICS Business Transaction Services (BTS) process. There can be many different user IDs and terminals that are associated with the transactions that are included in this affinity group.
LINK3270	All instances of all transactions in the group are associated with the same Link3270 bridge facility.

LUname	A group of transactions in which 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 LUname relations can be “pseudoconversation,” “logon,” “pseudoconversational,” “system,” or “permanent.”
User ID	A group of transactions in which 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 “pseudoconversation,” “signon,” “system,” or “permanent.”

9.1.4 Affinity lifetimes

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

System	The affinity lasts for as long as the target region exists and ends whenever the target region terminates (at a normal, immediate, or abnormal termination). (The resource that the transactions share that takes part in the affinity is not recoverable across CICS restarts).
Permanent	The affinity extends across all CICS restarts. (The resource that the transactions share that takes part in the affinity is recoverable across CICS restarts). This is the most restrictive of all of the inter-transaction affinities.
Process	The affinity exists until the BTS process completes.
Activity	The affinity exists until the BTS activity completes.
Facility	The affinity exists until the Link3270 bridge facility is deleted.
Pseudoconversation	The (LUname or user ID) affinity lasts for the whole pseudoconversation and ends when the pseudoconversation ends at the terminal.
Logon	The (LUname) affinity lasts for as long as the terminal remains logged on to CICS and ends when the terminal logs off.
Signon	The (user ID) affinity lasts for as long as the user is signed on and ends when the user signs off. This lifetime is only possible in those situations where only one user ID is permitted. Signon lifetime cannot be detected if multiple

users are permitted to be signed on with the same user ID at the same time.

Notes: If an affinity is both “user ID” and “LUsername” (all instances of all transactions in the group were initiated from the same terminal and by the same user ID), “LUsername” takes precedence.

9.2 The Affinities Scanner and the Affinities Collector

The *CICS IA Affinities Collector/Scanner* is designed to detect causes of inter-transaction affinity and transaction-system affinity for those who plan to use CICS workload management. The utility can be used to:

- ▶ Detect programs that use EXEC CICS commands that *might* cause transaction affinity.
- ▶ Create a file that contains combined affinity transaction group definitions that are suitable for input to CICSplex SM.

The utility determines the affinities that apply to a single CICS region, that is, a single-pure AOR or single combined TOR/AOR. It can be run against production CICS regions and is also useful in a test environment to detect possible affinities that new or changed application suites or packages introduce.

The utility is comprised of four main components:

- ▶ Affinities Scanner (batch)
- ▶ Affinities Collector (real-time)
- ▶ Affinities Reporter (batch)
- ▶ Builder (batch)

9.2.1 The Affinities Scanner

The Affinities Scanner is a batch utility that you can use to scan a load module library to detect those programs that issue the EXEC CICS commands that *might* cause transaction affinity. It examines the individual object programs, and looks for patterns that match the argument zero¹. The Affinities Scanner can indicate an affinity problem that does not really exist because a bit pattern that is found accidentally matches the argument zero format for an affinity command.

¹ Argument zero: When an EXEC CICS command is translated and compiled, it results in an encoded parameter list to be used with a call statement. The first parameter in this list is a constant known as the CICS argument zero. The first 2 bytes of this constant identify the command; for example, X'0A04' identifies it as a READQ TS command.

The Affinities Scanner detects the use of EXEC CICS commands, which we list in 9.1, “Overview of transaction affinities” on page 204.

The Affinities Scanner is independent of the language that the scanned program was written in and the release of CICS that the scanned program was translated under.

9.2.2 The Affinities Collector

The Affinities Collector can be used in real-time to detect *potential* transaction affinities in a running CICS region and saves details of the affinities in a z/OS data space. This data is then extracted to a number of VSAM files.

The Affinities Collector is comprised of the following items:

- ▶ Control transaction (CINT)
- ▶ Autosave transaction (CINB)
- ▶ Global exit programs
- ▶ Task related user exit program

The data is collected by the global user exit programs at exit points XEIOU, XBADEACT, XFAINTM, XMEOUT, XICEXP, and a TRUE at task start and task end. Between them, these exit programs intercept all of the EXEC CICS commands and other events (pseudoconversation end, terminal logoff, user signoff) that are needed to deduce the affinities and their relations and life times.

The Affinities Collector detects the EXEC CICS commands that are listed in Table 9-1 on page 211.

The Affinities Collector also detects:

- ▶ The end of pseudoconversations by detecting when one of the transactions in the pseudoconversation terminates without issuing an EXEC CICS RETURN TRANSID command with a non-zero transaction identifier.
- ▶ Log offs and sign offs by intercepting messages DFHSN1200, DFHZC3462, and DFHZC5966.
- ▶ Completion of CICS BTS activities and processes.

Worsening of transaction affinities relations

In some cases, the Affinities Collector might not detect enough occurrences (at least 10) of an affinity command to be sure that the affinity is definitely with a terminal (LUNAME), user ID (USERID), or CICS BTS process (BAPPL). In such cases, the Affinities Collector records the (worsened) affinity relation as GLOBAL instead of LUNAME or USERID. The Affinities Collector flags such relation worsening and the Affinities Reporter reports it.

Worsening of transaction affinities lifetimes

If a pseudoconversation ends, and the resource still exists, the Affinities Collector deduces that the lifetime is longer than PCONV: one of LOGON, SIGNON, SYSTEM, or PERMANENT.

If a log off or sign off occurs, and the resource still exists, the Affinities Collector deduces that the lifetime is longer than LOGON or SIGNON: either SYSTEM or PERMANENT.

If a CICS BTS activity completes and the resource still exists, the lifetime is worsened to process. If a CICS BTS process completes and the resource still exists, the lifetime is worsened to the system.

In some cases, the Affinities Collector might not detect a log off or sign off; therefore, we cannot be sure that the affinity lifetime is LOGON or SIGNON. In such cases, the Affinities Collector records the (worsened) lifetime as SYSTEM or PERMANENT instead of LOGON or SIGNON, for example, this occurs when the CICS region that the Affinities Collector is running on is a target region because it is impossible, in some cases, to detect a log off or sign off that occurs in a connected routing region. The Affinities Collector flags such relation worsening and the Affinities Reporter reports it.

Controlling the Affinities Collector

The Affinities Collector is monitored and controlled through the CINT transaction, which enables you to start, pause, continue, and stop the collection of affinity data into the tables in the data space.

The affinity data that the Affinities Collector collects is saved to the VSAM files by transaction CINB. This transaction saves data:

- ▶ When the Affinities Collector is stopped
- ▶ When the Affinities Collector is paused
- ▶ On a pre-determined time/activity basis

9.2.3 The Affinities Reporter

The Affinities Reporter is a batch utility that you can use to convert the affinity data that is collected by the Affinities Collector into two output formats:

- ▶ A report that presents the affinity data in a readable form that is suitable for system programmers and application designers. The report indicates the transactions and programs that issue EXEC CICS commands that cause inter-transaction and transaction-system affinities. This information helps to understand the transactions that make up the workload and can help to determine the changes that are needed to remove unwanted affinities.
- ▶ A file containing a set of basic affinity transaction group definitions in a syntax approximating to the batch API of CICSplex SM. This is intended as input to the Builder.

9.2.4 The Builder

The Builder is a batch utility that you can run against a set of files that contain the basic affinity transaction groups that the Affinities Reporter created. The output of the builder is a file that contains combined affinity transaction group definitions, as required by CICSplex SM, which insists that a given transaction can only appear in a single transaction group.

9.2.5 Detected API and SPI commands

In this section, we list the Affinity-related EXEC CICS commands that the Affinities Collector and the Affinities Scanner detects. All of the commands that we list in Table 9-1 on page 211 are capable of causing affinities; however, they might or might not actually do so. In Table 9-1 on page 211:

- ▶ The left column shows the CICS API commands that might create inter-transaction affinities.
- ▶ The center column shows the CICS API commands that might create transaction-system affinities.
- ▶ The right column shows the CICS SPI commands that might create transaction-system affinities.

Table 9-1 Affinity-related CICS API and SPI commands that the Affinities Collector and the Affinities Scanner collect

CICS API commands that might create inter-transaction affinities	CICS API commands that might create transaction-system affinities	CICS SPI commands that might create transaction-system affinities
ENQ DEQ READQ TS WRITEQ TS DELETEQ TS ADDRESS CWA LOAD RELEASE GETMAIN SHARED FREEMAIN RETRIEVE WAIT DELAY POST START CANCEL COLLECT STATISTICS	STARTBROWSE ACTIVITY STARTBROWSE CONTAINER STARTBROWSE EVENT STARTBROWSE PROCESS GETNEXT ACTIVITY GETNEXT CONTAINER GETNEXT EVENT GETNEXT PROCESS ENDBROWSE ACTIVITY ENDBROWSE CONTAINER ENDBROWSE EVENT ENDBROWSE PROCESS WAIT EXTERNAL WAIT EVENT WAITCICS	ENABLE PROGRAM DISABLE PROGRAM EXTRACT EXIT INQUIRE SET PERFORM RESYNC DISCARD CREATE

With the detection of the API and SPI commands:

- ▶ The Affinities Scanner might detect some instances of these commands that do not cause an affinity, for example, all FREEMAIN commands are detected but only those that are used to free GETMAIN SHARED storage can cause an affinity.
- ▶ The Affinities Scanner also detects MVS POST SVC calls and MVS POST LINKAGE=SYSTEM non-SVC calls because of their relationship to the various EXEC CICS WAIT commands.
- ▶ The Affinities Collector does not search for transient data and file control EXEC CICS commands. They are assumed not to cause affinities because you can define transient data and file control resources as remote in which case the request is function-shipped, which causes no affinity problem.
- ▶ The Affinities Collector ignores commands that target remote resources and are function shipped because function-shipped commands do not cause affinity problems in the region where the Collector is running.
- ▶ The Affinities Collector and the Affinities Scanner do not search for commands that are issued by any program named CAUxxxxx, CIUxxxxx, or DFHxxxxx because CICS programs are not considered part of the workload.

Also, the Affinities Collector does not search for commands that are issued from:

- DB2 and DBCTL task-related user exits
- User-replaceable programs
- ▶ There are other ways in which transactions can cause affinities with each other; however, they are not readily detectable by the Affinities Collector because they do not take place through the EXEC CICS API.
- ▶ The Affinities Collector lists WAIT commands as transaction-system affinities because only half of the affinity can be detected. (The Collector does not detect MVS POST calls or the hand-posting of ECBs.)
- ▶ The Affinities Reporter ignores ENQ and DEQ commands that specify an ENQSCOPE name.

9.3 Customizing and running the Affinities Scanner

To detect as many affinities as possible, we recommend that you first use the Affinities Scanner component. Just running the Affinities Collector component might not be enough because it might be difficult to ensure that all parts of the workload, especially the rarely used transactions, were used while the Affinities Collector was active.

More details about how to use the Affinities Scanner component are in Chapter 3, “Scanner component” on page 53.

9.3.1 The summary report

To get an overview of the modules with potential transaction affinities we used the JCL in Example 9-1.

Example 9-1 Affinities Scanner JCL CIUJCLLS

```
//SCAN      EXEC PGM=CIULMS,PARM='$SUMMARY'  
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,  
//          DISP=SHR  
//          DD DSN=CIU.V2R2M0.SCIULODE,  
//          DISP=SHR  
//INPUT     DD DSN=CIU.TEST.LOADLIB,  
//          DISP=SHR  
//SYSPRINT DD SYSOUT=*  
//INTMOD    DD DUMMY  
//DETAIL    DD DUMMY
```

The first part of the summary report shows a summary listing of all the modules in the scanned library, as shown in Figure 9-1.

```

CICS INTERDEPENDENCY ANALYZER Version 2.2.0                                04/24/08   Page
1
LOAD MODULE SCANNER - SUMMARY LISTING OF CIU.TEST.LOADLIB

Module      Module      Module      Language  Possible statements.....  Comment
Name        Length      Language    Version   Affinities  Dependencies  MVS POSTs
-----
AFFTESTA    00001758    COBOL II    LE        5           5           0
AFFTESTB    00001778    COBOL II    LE        4           4           0
AFFTESTC    00001678    COBOL II    LE        2           2           0
AFFTESTD    00001838    COBOL II    LE        2           2           0
AFFTESTE    00001768    COBOL II    LE        2           2           0
AFFTESTS    00003B68    COBOL II    LE        24          37          0
CALDTLIC    000028C0                                0           0           0

```

Figure 9-1 Affinities Scanner Report: Listing of modules

At the end of the report we see the statistics for the scanned library, as shown in Figure 9-2.

```

CICS INTERDEPENDENCY ANALYZER Version 2.2.0                                04/24/08   Page
4
LOAD MODULE SCANNER - SUMMARY LISTING OF CIU.TEST.LOADLIB

LOAD LIBRARY STATISTICS
=====
Total modules in library                = 110
Total modules scanned                  = 109
Total CICS modules/tables (not scanned) = 1
Total modules in error (not scanned)   = 0
Total modules containing possible MVS POSTs = 0
Total modules containing possible Dependency commands = 78
Total modules containing possible Affinity commands = 49
  Total ASSEMBLER modules              = 24
  Total C/370 modules                  = 0
  Total COBOL modules                  = 1
  Total COBOL II modules               = 52
  Total PL/I modules                   = 7
Total number of possible Dependency commands = 565
Total number of possible Affinity commands = 307

```

Figure 9-2 Affinities Scanner Report: Load library statistics

9.4 Running the Affinities Collector

To start the Affinities Collector:

1. Use transaction CINT → panel CIU000.
2. On panel CIU000, select option 2 - Configure Region Options. Go to panel CIU200.
3. On panel CIU200, enter action code 6 next to the region in which you want to start the Affinities Collector. Go to panel CIU260.
4. On panel CIU260, enter Y for “Data to Collect”, as shown in Figure 9-3.

```
CIU260          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/24
                  General Options for                                02:50:05PM
                  CICS Sysid   : TS0A   CICS Applid   : IYCYZC3A

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

Control options (Fields may also be set to blanks for default)
Data to Collect . . . . . : Y   (Y=Affinity, N=Interdependency)
Perform periodic saves . . . :   (Y=Yes, N=No)
Trigger for CINB start . . . :   (2 to 9999 thousand record updates)
Inquire for DB2 resources. . :   (Y=Yes, N=No)
Restore data on start . . . :   (Y=Yes, N=No)
Multiple signon with same id :   (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 . . :   (1 to 8 characters)

CICS Sysid: TS0A   CICS Applid: IYCYZC3A   TermID: TC74
```

Figure 9-3 CIU620 - Options to start the Affinity Collector

For more information about the Affinities Collector, refer to Chapter 4, “The Collector” on page 73.

9.5 Customizing and running the Affinities Reporter

The Affinities Reporter is a batch utility that you can use to:

- ▶ Convert the affinity data into reports in a readable format.
- ▶ Create a file of affinity-transaction-group definitions in a syntax approximating to the batch API of CICSplex SM. This file is intended as input to the Builder component.

The Affinities Reporter can take as input:

- ▶ The VSAM files from the Affinities Collector (sample job CIUAFFRP)
- ▶ The Affinity tables from the DB2 database (sample job CIUAFFRD)

The output is the same in both cases. Refer to the *CICS Interdependency Analyzer User's Guide and Reference*, SC34-6790.

We used the job CIUAFFRP and ran it against the files with DD names CAUAFF1, CAUAFF2, and CAUAFF3. These VSAM files contain the collected data from the Affinities Collector. The Affinities Reporter creates the new data set TRANGRPS, which is the input file for the Builder.

If you do not specify any affinity types on the CMDGRPS DD statement, or specify CMDGRPS DD DUMMY, all affinity types are selected for reporting. The first part of the report lists the selected affinity types.

The TRANGRPS DD statement specifies the name of the data set where the basic affinity transaction groups are to be sent.

The Affinities Reporter cannot read records from the CAUAFF1, CAUAFF2, and CAUAFF3 VSAM files while the Affinities Detector has those files opened for update. Therefore, do not run the Affinities Reporter at the same time that you run the Affinities Detector.

We used the JCL in Example 9-2.

Example 9-2 Affinities Reporter JCL CIUAFFRP

```
//REPORT EXEC PGM=CIUAFFR1,PARM='WORSEN=YES'  
//STEPLIB DD DSN=CIU.V2R2M0.SCIULOAD,  
// DISP=SHR  
// DD DSN=CIU.V2R2M0.SCIULODE,  
// DISP=SHR  
//CIUAFF1 DD DSN=CICSTLS.IAV22.CIUAFF1,  
// DISP=SHR  
//CIUAFF2 DD DSN=CICSTLS.IAV22.CIUAFF2,  
// DISP=SHR
```

```

//CIUAFF3 DD DSN=CICSTLS.IAV22.CIUAFF3,
//          DISP=SHR
//CIUCNTL DD DSN=CICSTLS.IAV22.CIUCNTL,
//          DISP=SHR
/** Enter ALL for all APPLIDs or a list of APPLIDS to be loaded
/** No entry is treated the same as ALL
//APPLID DD *
/*
//CMDGRPS DD *
/*
//TRANGRPS DD DSN=CIUTEST.AFFGRP1,
//          DISP=(NEW,CATLG,DELETE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000),
//          SPACE=(TRK,(2,2),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*

```

We used PARM='WORSEN=YES' to specify that the Affinities Reporter is to worsen transaction affinity relations for those affinities where the Affinities Detector has not detected at least 19 occurrences. For more information, refer to "Worsening of transaction affinities relations" on page 208.

For each affinity type that is specified on the CMDGRPS DD statement, the Affinities Reporter outputs each individual affinity both in report format and as definitions for basic affinity-transaction-groups that are suitable for input to the Builder.

Figure 9-4 on page 217 shows the first part of the affinity reports that lists the selected affinity types.

Generated by CICS Transaction Inter-dependency Utility (AFFINITY Reporter) on 2008/04/23
Note: NOT suitable for input to CICSplex SM

--LIST OF CICS COMMAND GROUPS--

2008/04/23 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 220 - Page: 1
--LIST OF CICS COMMAND GROUPS--

Affinity Type	Reporting
-----	-----
Inter-Transaction Affinities	

CWA	Yes
CANCEL	Yes
ENQ	Yes
GETMAIN	Yes
LOAD	Yes
RETRIEVE	Yes
TS	Yes
Transaction-System Affinities	

COLLECT	Yes
DISCARD	Yes
ENABLE	Yes
EXTRACT	Yes
INQUIRE	Yes
PERFORM	Yes
RESYNC	Yes
WAIT	Yes
CREATE	Yes

Figure 9-4 Affinity Reporter - selected affinity types

The second part of the report, Figure 9-5 on page 218, shows the inter-transaction affinities.

Trangroup : CW.00000001
 Affinity : GLOBAL
 Lifetime : SYSTEM

Tranid	Program	Offset	Usage	Command	Terminal	CBTS Task	Link3270
AFTA	AFFTESTA	0000068E	2	ADDRESS CWA	Yes	No	No
AFTS	AFFTESTS	00000C66	1	ADDRESS CWA	Yes	No	No
Total Transactions			:	2			
Total Programs			:	2			

Trangroup : EQ.00000001
 Affinity : GLOBAL
 Lifetime : SYSTEM
 Resource : A(11809DB0)
 Length : 4

Tranid	Program	Offset	Usage	Command	Terminal	CBTS Task	Link3270
AFTS	AFFTESTS	00000F70	1	ENQ ADDRESS	Yes	No	No
AFTS	AFFTESTS	00000FB4	1	DEQ ADDRESS	Yes	No	No
Total Transactions			:	1			
Total Programs			:	1			

Trangroup : EQ.00000002
 Affinity : GLOBAL
 Lifetime : SYSTEM
 Resource : A(11809E38)
 Length : 4

Tranid	Program	Offset	Usage	Command	Terminal	CBTS Task	Link3270
AFTA	AFFTESTA	0000077E	2	ENQ ADDRESS	Yes	No	No
AFTA	AFFTESTA	000007B8	2	DEQ ADDRESS	Yes	No	No
Total Transactions			:	1			
Total Programs			:	1			

Trangroup : EQ.00000003
 Affinity : GLOBAL
 Lifetime : SYSTEM
 Resource : 'A_C_E_GHIJKLMNOPQRSTUVWXYZ'
 (C16DC36DC56DC7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9)
 Length : 26

Tranid	Program	Offset	Usage	Command	Terminal	CBTS Task	Link3270
AFTS	AFFTESTS	00000D16	1	ENQ NAME	Yes	No	No
AFTS	AFFTESTS	00000D62	1	DEQ NAME	Yes	No	No
Total Transactions			:	1			
Total Programs			:	1			

Figure 9-5 Affinity Reporter - inter-transaction affinities

Finally, in the third part of the report, Figure 9-6, the transaction-system affinities are shown.

```

2008/04/23 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 220 - Page: 9
TRANSACTION-SYSTEM AFFINITIES REPORT FOR APPLID: IYCYZC3A - DISCARD COMMANDS

```

Tranid	Program	Offset	Usage	Command
AFTS	AFFTESTS	0000167A	1	DISCARD FILE
AFTS	AFFTESTS	000018F0	1	DISCARD FILE
Total Transactions			:	1
Total Programs			:	1

```

2008/04/23 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 220 - Page: 10
TRANSACTION-SYSTEM AFFINITIES REPORT FOR APPLID: IYCYZC3A - INQUIRE/BROWSE/SET COMMANDS

```

Tranid	Program	Offset	Usage	Command
AFTA	AFFTESTA	00000744	2	INQUIRE PROGRAM
AFTS	AFFTESTS	000010F4	1	INQUIRE TSQNAME
AFTS	AFFTESTS	000015E6	1	INQUIRE FILE
AFTS	AFFTESTS	00001630	1	SET FILE
AFTS	AFFTESTS	000016B6	1	INQUIRE FILE
AFTS	AFFTESTS	00001770	132	INQUIRE FILE
AFTS	AFFTESTS	000017B4	1	INQUIRE FILE
AFTS	AFFTESTS	0000185C	1	INQUIRE PROGRAM
AFTS	AFFTESTS	000018A6	1	SET FILE
AFTS	AFFTESTS	0000192C	1	INQUIRE PROGRAM
AFTS	AFFTESTS	00001986	3,017	INQUIRE PROGRAM
AFTS	AFFTESTS	000019CA	1	INQUIRE PROGRAM
Total Transactions			:	2
Total Programs			:	2

```

2008/04/23 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 220 - Page: 11
TRANSACTION-SYSTEM AFFINITIES REPORT FOR APPLID: IYCYZC3A - CREATE COMMANDS

```

Tranid	Program	Offset	Usage	Command
AFTS	AFFTESTS	0000159C	1	CREATE FILE
AFTS	AFFTESTS	00001812	1	CREATE PROGRAM
Total Transactions			:	1
Total Programs			:	1

Figure 9-6 Affinity Reporter - transaction-system affinities

The output that we shown in Figure 9-7 on page 220 was created in the new file TRANGRPS is used as input for the Builder.

Transaction-system affinities do not appear because they are not of interest to a dynamic routing program, neither, for the same reason, do transactions that were not initiated from a terminal. The same applies to BTS and Link3270 bridge mechanism.

Workload separation in a CICSplex SM environment ensures correct processing of transaction-system affinities. With workload separation, a group of

transactions are associated with a group of target regions. If only one target region can be associated with a group of transactions, this region is a single point of failure.

```

* HEADER APPLID(IYCYZC3A) SAVEDATE(20080423) SAVETIME(132319 );
*
* Generated by CICS Transaction Affinities Utility (Reporter) on 2008/04/23
* Note: NOT suitable for input to CICSplex SM
*
CREATE TRANGRP NAME(CW.00000001) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(ADDRESS CWA );
CREATE DTRINGRP TRANGRP(CW.00000001) TRANID(AFTA);
CREATE DTRINGRP TRANGRP(CW.00000001) TRANID(AFTS);
*
CREATE TRANGRP NAME(EQ.00000001) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(L=4 A=11809DB0 );
CREATE DTRINGRP TRANGRP(EQ.00000001) TRANID(AFTS);
*
CREATE TRANGRP NAME(EQ.00000002) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(L=4 A=11809E38 );
CREATE DTRINGRP TRANGRP(EQ.00000002) TRANID(AFTA);
*
CREATE TRANGRP NAME(EQ.00000003) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(L=26 A_C_E_GHIJKLMNOPQRSTUVWXYZ);
CREATE DTRINGRP TRANGRP(EQ.00000003) TRANID(AFTS);
*
CREATE TRANGRP NAME(EQ.00000004) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(L=55 ABCDEFGHIJKLMNOPQRSTUVWXYZ);
CREATE DTRINGRP TRANGRP(EQ.00000004) TRANID(AFTS);
*
CREATE TRANGRP NAME(EQ.00000005) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(L=210 A23456789012345678901234);
CREATE DTRINGRP TRANGRP(EQ.00000005) TRANID(AFTE);
*
CREATE TRANGRP NAME(RW.00000001) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(RW.AFTC C1C6E3C3 );
CREATE DTRINGRP TRANGRP(RW.00000001) TRANID(AFTB);
*
CREATE TRANGRP NAME(TS.00000001) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
DESC(TS.LONGTSQ NAMES... D3D6D5C7);
CREATE DTRINGRP TRANGRP(TS.00000001) TRANID(AFTB);
* HEADER APPLID(IYCYZC37) SAVEDATE( ) SAVETIME( );
*
* Generated by CICS Transaction Affinities Utility (Reporter) on 2008/04/23
* Note: NOT suitable for input to CICSplex SM

```

Figure 9-7 Affinity Reporter - contents of file TRANGRPS

9.5.1 Tips for analyzing reports

Sometimes the report that the Affinities Reporter produces can contain results that appear odd at first glance, for example:

► **COBOL affinities:**

- If an application program is invoked using the CALL statement, CICS COBOL runtime code issues an EXEC CICS LOAD HOLD for the program and branches to it directly, which only causes an affinity if the program is not re-entrant, that is, it modifies itself; otherwise, there is no affinity.

- CICS COBOL run-time code writes data to a TS queue if an abend occurs. The TS queue name used is 'CEBR' plus the termed of the terminal, or blanks if no terminal. This does not cause an affinity.

► **LOGON or SYSTEM when PCONV expected.**

When you deal with an application that is known to use TS queues within a pseudo-conversation, but never beyond, there might be occurrences in the CICS IA report of affinity groups that appear as LUNAME/SYSTEM or LUNAME/LOGON, instead of the expected LUNAME/PCONV:

- A SYSTEM lifetime occurs if the installation uses a session manager that logs users off after a predetermined quiet time. When the logoff occurs in the middle of a pseudo-conversation, CICSIA notices that the TS Queue still exists and increases the lifetime to SYSTEM.
- A LOGON lifetime is when the user switches off the terminal in the middle of a pseudo-conversation and causes a VTAM® line error, which causes an error transaction to be attached internally at the terminal. CICSIA notices that the TS Queue exists at the end of that transaction and increases the lifetime to LOGON.

In both of these circumstances, the real lifetime is PCONV because although the TS queue exists at the end of the pseudo-conversation, the data in it is never used again. Normally the first action of a new pseudo-conversation is to delete the contents of all such TS queues for that terminal.

Consider making the following modifications to the affinity transaction groups before inputting them to the Builder:

► **Remove false affinities**

False affinities can arise because sharing resources occurs on a read-only basis, so it is possible for the resource to be replicated across cloned CICS regions. The prime example of this is a read-only CWA, where the CWA is set up at CICS startup (for example, from a PLTPI program) and only read thereafter. (An alternative way of removing this false affinity is to prohibit detection of ADDRESS CWA by the Affinities Detector using the CINT options.)

► **Verify affinity relation worsening**

An affinity that has a relation of LUNAME, BAPPL, or USERID might be worsened to GLOBAL because the Affinities Detector has not seen enough examples of the affinity to be convinced that it is related to a terminal or user ID. Change this to LUNAME or USERID (and correct the lifetime) if you know that the affinity really is terminal or user ID-related. You might want to prevent worsening by specifying PARM='WORSEN=NO' for the Affinities Reporter.

- ▶ Verify affinity lifetime worsening

A LUNAME affinity with a lifetime of LOGON or a USERID affinity with a lifetime of SIGNON, can be worsened to SYSTEM or PERMANENT because the Affinities Detector cannot always observe log offs or sign offs. Change this to LOGON or SIGNON if you know that to be the correct lifetime.
- ▶ Consider changing LUNAME affinity relation to USERID

A LUNAME affinity group can be both LUNAME and USERID because all instances of all transactions in the group were initiated from the same terminal by the same user ID. This appears in the report as LUNAME because LUNAME takes precedence. If you know that the affinity is primarily user ID-related, change the affinity to USERID. (This might be indicated by other similar affinity groups that appear in the report with USERID).
- ▶ Add WAIT affinities

The Affinity Reporter reports the use of WAIT EVENT, WAITCICS, and WAIT EXTERNAL commands as transaction-system affinities because the Affinities Detector cannot detect the corresponding posting of the ECBs being waited upon. Identify the posting transactions, and create affinity transaction groups to describe the affinities. The output from the Scanner might be particularly useful here because it finds programs that issue MVS POST commands.
- ▶ Add other affinities

Scanner output and your knowledge of your applications can identify additional affinities. Create affinity transaction groups to describe them.
- ▶ Add GETMAIN storage sharers

The Detector cannot detect transactions that share storage other than through EXEC CICS commands. Although it detects GETMAIN SHARED/FREEMAIN affinities, the address of the storage might have been passed to a third transaction. Add such transactions to the affinity transaction group.

9.5.2 How to proceed

We cannot give a general recommendation on how to proceed. Every situation is different and it mostly depends on the number of affinities and on how difficult it is to remove the affinities.

Our customers have taken different approaches on how to proceed. The most common are:

- ▶ Minor clean-up of affinities

In this situation, the customer decided not to change the applications. They analyzed the affinities report and cleaned-up the TRANGRP file as

appropriate. They then used the Builder component to create transaction group definitions that were suitable for CICSplex SM and applied the CICSplex SM definitions. With this approach, they could implement CICSplex SM workload management with little effort and in a brief period of time. However, they also discovered that with this approach, the dynamic workload management, was not as effective as expected because of all the affinities that CICSplex SM had to consider. Therefore, they decided to review and change the applications to remove affinities wherever possible.

► Application changes

One of our customers decided to review and change their applications first and remove as many affinities as possible. In this case, the up front investment in time and costs was bigger, but they could take advantage of the benefits of CICSplex SM workload management early on.

9.6 Customizing and running the Builder

The Builder runs as a batch job to build affinity transaction groups that are suitable for input to CICSplex SM.

The Builder takes, as input, a set of files that contain basic affinity transaction groups, combines those groups, and produces a file that contains combined affinity transaction groups. CICSplex SM requires that a transaction identifier is in one transaction group only, and the Builder satisfies this by combining groups that contain the same transaction identifier.

Note: You can use the Reporter to produce files of basic transaction affinity groups for input to the Builder. The files can be from several runs of the Detector (for example, against a production CICS region and a test CICS region) but must be for the same workload.

We used the JCL in Figure 9-3 on page 214.

Example 9-3 Builder JCL CIUAFFBL

```
//BUILD EXEC PGM=CIUBLD,  
//      PARM=('STATE=ACTIVE,MATCH=LUNAME,DSPSIZE=16',  
//          'CONTEXT=TOOLPLX1')  
//STEPLIB DD DSN=CIU.V2R2MO.SCIULOAD,  
//          DISP=SHR  
//          DD DSN=CIU.V2R2MO.SCIULODE,  
//          DISP=SHR  
//REPGRPS DD DSN=CIUTEST.AFFGRP1,
```

```

//          DISP=SHR
//AFFGRPS DD DSN=CIUTEST.CPSMGRPS(IYCYZC3A),
//          DISP=(NEW,CATLG,DELETE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000),
//          SPACE=(CYL,(2,2,2))
//APPLID   DD *
/*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*

```

The output of the Builder program, CIUABLD, is a report and a data set (AFFGRPS DD statement) that contains the combined affinity transaction groups.

The first part of the report is the Data set processed report, shown in Figure 9-8, which shows the names of all of the input data sets (specified on the REPGRPS DD statement) that were read.

2008/04/29 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 220 - Page: 1			
BUILDER REPGRPS DATASETS PROCESSED REPORT			
Dataset Name	CICS APPLID	Detector Last Save Date	Detector Last Save Time
-----	-----	-----	-----
CIUTEST.AFFGRP1	IYCYZC3A	20080423	132319

Figure 9-8 Builder - Data set processed report

The second part of the report usually is the Group merge report, shown in Figure 9-9 on page 225. This report can serve as an audit trail that you can use to establish which basic group caused a severe worsening of an affinity lifetime.

An Empty transaction group report, as part of the report, shows transaction groups that were defined but contain no transactions. Empty transaction groups indicate that you made a mistake. The Affinity Reporter cannot produce empty transaction groups; therefore, you must create the input by hand and probably omit some corresponding CREATE DTRINGRP statements.

Trangroup : AFTAGRP
Affinity : GLOBAL
Lifetime : SYSTEM
Match : LUNAME
State : ACTIVE

Consists of Transactions
AFTA AFTS

Consists of groups merged from
CIUTEST.AFFGRP1

CW.00000001 (GLOBAL SYSTEM)	EQ.00000001 (GLOBAL SYSTEM)	EQ.00000002 (GLOBAL SYSTEM)
EQ.00000003 (GLOBAL SYSTEM)	EQ.00000004 (GLOBAL SYSTEM)	

Trangroup : AFTBGRP
Affinity : GLOBAL
Lifetime : SYSTEM
Match : LUNAME
State : ACTIVE

Consists of Transactions
AFTB

Consists of groups merged from
CIUTEST.AFFGRP1

RW.00000001 (GLOBAL SYSTEM)	TS.00000001 (GLOBAL SYSTEM)
------------------------------	------------------------------

Trangroup : AFTEGRP
Affinity : GLOBAL
Lifetime : SYSTEM
Match : LUNAME
State : ACTIVE

Consists of Transactions
AFTE

Consists of groups merged from
CIUTEST.AFFGRP1

EQ.00000005 (GLOBAL SYSTEM)

Figure 9-9 Builder - Group merge report

The data set (with AFFGRPS DD statement) contains the combined affinity transaction group. This data set is suitable for input to CICSplex SM using the BATCHREP utility. Figure 9-10 on page 226 shows the contents of the file.

```

* HEADER APPLID(BUILDER ) SAVEDATE(20080423) SAVETIME(132319 );
*
* Generated by CICS IA Transaction Affinities (Builder) on 2008/04/29
* Note: Suitable for input to CICSplex SM
*
CONTEXT TOOLPLX1;
*
* REMOVE TRANGRP NAME(AFTAGRP );
CREATE TRANGRP NAME(AFTAGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
      AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
  CREATE DTRINGRP TRANGRP(AFTAGRP ) TRANID(AFTA);
  CREATE DTRINGRP TRANGRP(AFTAGRP ) TRANID(AFTS);
*
* REMOVE TRANGRP NAME(AFTBGRP );
CREATE TRANGRP NAME(AFTBGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
      AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
  CREATE DTRINGRP TRANGRP(AFTBGRP ) TRANID(AFTB);
*
* REMOVE TRANGRP NAME(AFTEGRP );
CREATE TRANGRP NAME(AFTEGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
      AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
  CREATE DTRINGRP TRANGRP(AFTEGRP ) TRANID(AFTE);

```

Figure 9-10 Builder - File with combined affinity transaction groups

9.7 Uploading the data into CICSplex SM

To upload the file that we created in the previous step, we use the CICSplex SM tool BATCHREP (batched repository update facility).

To submit a BATCHREP job, you can either use the CICSplex SM Web user interface (Administration views → Batched repository update requests) or a batch job. Example 9-4 shows an example of a BATCHREP job.

Example 9-4 BATCHREP JCL sample

```

//BTCHUPD EXEC PGM=EYU9XDBC,REGION=2048K
//STEPLIB DD DSN=CTS310.CPSM310.SEYUAUTH,DISP=SHR
//          DD DSN=CTS310.CPSM310.SEYULOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN   DD *
          CMASNAME(CMAS04)

```

```
EXECUTE
INPUTDSN(CIUTEST.CPSMGRPS)
INPUTMEMBER(IYCYZC3A)
OUTPUTUSER(CIUTEST)
PRINTNODE(LOCAL)
```

To start the BATCHREP by using the CICSPlex SM Web user interface:

1. From the main menu, select **Administration views**.
2. In the Administration view, select **Batched repository update job** (last item in General views).
3. In the Batched repository update job, select the first record and press **Select**.
4. In the **Execute** view, enter the values in Figure 9-11.

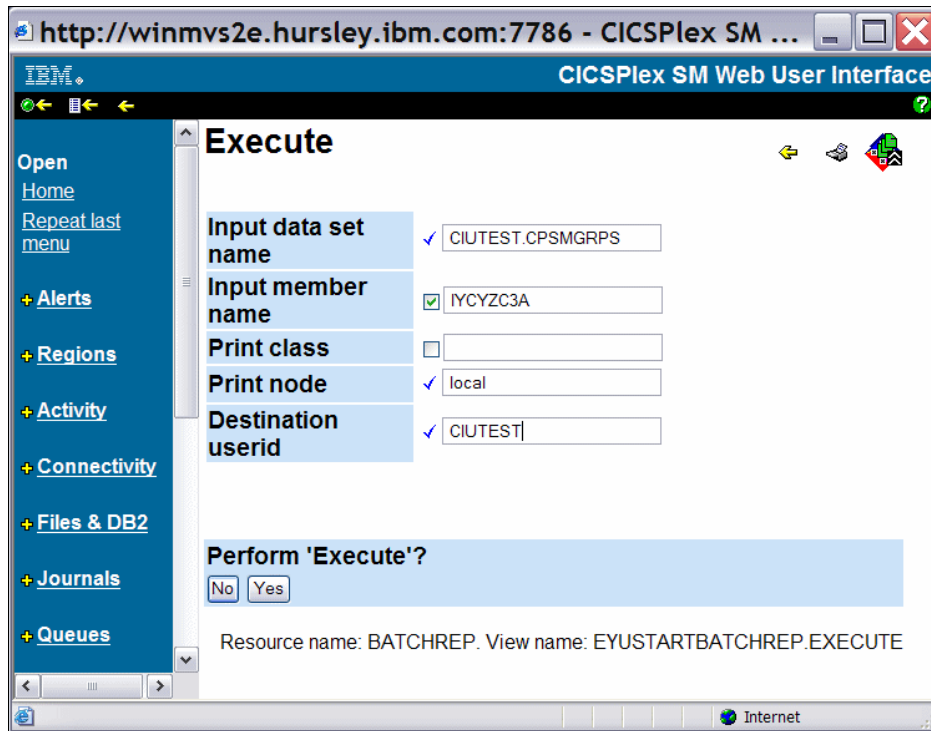


Figure 9-11 BATCHREP - WUI input parameters

Note: The Print class, Print node, and Destination user ID values are site-specific. Consult your z/OS administrator for valid values for these fields. Be aware, however, that the Print Class value should identify a HELD output class so that the results of the batch run are validated.

You can check the result of the batched repository update facility:

- ▶ In EYU9LOG
- ▶ In the output spool file with the provided Destination user ID
- ▶ With the CPSM Web interface

Example 9-5 shows the result of our submitted BATCHREP in EYU9LOG.

Example 9-5 EYU9LOG - result of BATCHREP

EYULOG - CONSOLIDATED MESSAGE LOG

```
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(NONE) MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(TRAN) MINOR_NAME(AFTA) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(TRAN) MINOR_NAME(AFTS) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTBGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(NONE) MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTBGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(TRAN) MINOR_NAME(AFTB) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTEGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(NONE) MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTEGRP) MAJOR_VER(NONE)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 MINOR_ID(TRAN) MINOR_NAME(AFTE) MINOR_VER(NONE) By User(CIUTEST) On
System(CMAS04)
04/30/2008 13:08:25 EYUXD0002I IYCYZC30 Date(12108) Time(13:08:25)
```

Example 9-6 shows the results of the spool output file.

Example 9-6 Spool output file - result of BATCHREP

```
CICSplex SM - Repository Process Report                                WED 30 APR 08 13:08:25 Page      1
Input DSN: CIUTEST.CPSMGRPS                                         Input Member: IYCYZC3A      Run Type: EXECUTE
* HEADER APPLID(BUILDER ) SAVEDATE(20080423) SAVETIME(132319 );
*
* Generated by CICS IA Transaction Affinities (Builder) on 2008/04/29
* Note: Suitable for input to CICSplex SM
*
CONTEXT TOOLPLX1;
*
* REMOVE TRANGRP NAME(AFTAGRP );
CREATE TRANGRP NAME(AFTAGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
      AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE) MINOR_ID(NONE)
EYUXD0002I IYCYZC30 MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
      CREATE DTRINGRP TRANGRP(AFTAGRP ) TRANID(AFTA);
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE) MINOR_ID(TRAN)
```

```

EYUXD0002I IYCYZC30 MINOR_NAME(AFTA) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
CREATE DTRINGRP TRANGRP(AFTAGRP ) TRANID(AFTS);
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTAGRP) MAJOR_VER(NONE) MINOR_ID(TRAN)
EYUXD0002I IYCYZC30 MINOR_NAME(AFTS) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
*
* REMOVE TRANGRP NAME(AFTBGRP );
CREATE TRANGRP NAME(AFTBGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTBGRP) MAJOR_VER(NONE) MINOR_ID(NONE)
EYUXD0002I IYCYZC30 MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
CREATE DTRINGRP TRANGRP(AFTBGRP ) TRANID(AFTB);
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTBGRP) MAJOR_VER(NONE) MINOR_ID(TRAN)
EYUXD0002I IYCYZC30 MINOR_NAME(AFTB) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
*
* REMOVE TRANGRP NAME(AFTEGRP );
CREATE TRANGRP NAME(AFTEGRP ) AFFINITY(GLOBAL ) AFFLIFE(SYSTEM )
AFFAUTO(YES) MATCH(LUNAME) STATE(ACTIVE );
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(TRANGRP) MAJOR_NAME(AFTEGRP) MAJOR_VER(NONE) MINOR_ID(NONE)
EYUXD0002I IYCYZC30 MINOR_NAME(NONE) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE
CREATE DTRINGRP TRANGRP(AFTEGRP ) TRANID(AFTE);
EYUXD0002I IYCYZC30 Add CONTEXT(TOOLPLX1) MAJOR_ID(DTRINGRP) MAJOR_NAME(AFTEGRP) MAJOR_VER(NONE) MINOR_ID(TRAN)
EYUXD0002I IYCYZC30 MINOR_NAME(AFTE) MINOR_VER(NONE) By User(CIUTEST) On System(CMAS04) Date(12108)
EYUXD0002I IYCYZC30 Time(13:08:25)
EYUXU0218I IYCYZC30 Batch CREATE request complete - Status(OK) - Associated fields NONE

```

Note: If you get the error message EYUXU0201E (like in Example 9-7 on page 229) the reason might be that the user ID of the CMAS job is not authorized to access the file. In this case, locate the RACF error message in the CMAS job log.

Example 9-7 shows an example of the error message.

Example 9-7 Error message

```

CICSplex SM - Repository Process Report                                WED 30 APR 08 10:14:46 Page 1
Input DSN: CIUTEST.CPSMGRPS                                         Input Member: IYCYZC3A   Run Type: CHECK
EYUXU0201E IYCYZC30 Failure opening input file (CIUTEST.CPSMGRPS) - return code (00000000)

```

Figure 9-12 on page 230 shows the result of when you use the CICSplex SM Web User Interface (Administration view → Workload manager administration views → Transactions in transaction groups).

http://winmvs2e.hursley.ibm.com:7786 - CICSplex SM Web User Interface - IY...

CICSplex SM Web User Interface

Dynamic transaction in transaction group

EYUVC1280I 4 records collected at 05/02/08 10:25:42.

Context: TOOLPLX1
 Transaction group name: [v] Automatic refresh: 60 seconds.
 Transaction name: [v] Aa [Refresh]

4 records on 1 pages.

Record	Transaction group name	Transaction name	Pseudo-conversational mode	Last time the definition was changed
1 <input type="checkbox"/>	AFTAGRP %	AFTA	N_a	04/30/08 13:08:25
2 <input type="checkbox"/>	AFTAGRP %	AFTS	N_a	04/30/08 13:08:25
3 <input type="checkbox"/>	AFTBGRP %	AFTB	N_a	04/30/08 13:08:25
4 <input type="checkbox"/>	AFTEGRP %	AFTE	N_a	04/30/08 13:08:25

4 records on 1 pages.

[Create...] [Remove...] [Map]

Resource name: DTRINGRP. View name: EYUSTARTDTRINGRP.TABULAR

Done Internet

Figure 9-12 CPSM WUI - result of BATCHREP



Hints and tips

In this chapter, we provide hints and tips for CICS IA. We include sections on the collector performance, DB2 performance, and DB2 table clean up.

10.1 Collector performance

In this section, we look at the CICS IA options that you can use to manage the performance impact of running CICS IA in your environments.

We only look at the collection of dependency resource information, but you can use the same options and principles to collect affinity resource information.

10.1.1 What resources to collect

First, decide which resources you want to collect information about, and ascertain why you want to collect this information. What is your goal in collecting information?

For dependency collection, there are four main resource types:

- ▶ CICS
- ▶ MQ
- ▶ DB2
- ▶ IMS (DLI calls only)

You can control the collection of DB2, MQ, and IMS from option 2 in the CINT transaction, for example, you might only have MQ in one of your CICS regions, so you then want to switch to MQ in all other regions.

Example: Collecting MQ resource information in one region

To collect MQ resource information in one region:

1. Select option 2 for Configure Region Options, as shown in Figure 10-1 on page 233.

```
CIU000          CICS Interdependency Analyzer for z/OS - V2R2M0      2008/04/23
                                     Main Administration Menu          09:56:36AM

Select one of the following.  Then press Enter.

2  1  Operations Menu.
   2  Configure Region Options.
   3  Configure Global Options.

CICS Sysid: Z328  CICS Applid: IYDZZ328  TermID: TC46

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

Figure 10-1 CINT: Select option 2 for Configure Region Options

2. From the Region configuration menu, select option 5 for DB2/IMS/MQ Options against the DEFAULT entry, as shown in Figure 10-2 on page 234.

```

CIU200          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/23
                  Region Configuration Menu                               10:00:23AM

Type action code then press ENTER.                                     More :
1=Add Region      3=Delete Region   5=DB2/IMS/MQ Options  7=Affinity Options
2=Copy Region    4=CICS Options     6=General Options   8=Time/Date Options

      CICS      CICS  New      New
Act  Applid   Sysid  Applid   Sysid  Status      Collecting
 5   DEFAULTS DFTS
     ALL      ALL
     IYDZZ138 Z138
     IYDZZ228 Z228
     IYDZZ238 Z238
     IYDZZ318 Z318
     IYDZZ328 Z328
                                     UNCONNECTED
                                     UNCONNECTED
                                     UNCONNECTED
                                     UNCONNECTED
                                     STOPPED

CICS Sysid: Z328  CICS Applid: IYDZZ328  TermID: TC46

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

```

Figure 10-2 CINT: Select option 5 against the DEFAULT

3. In the DB2/MQ/IMS resource option menu, set the MQ option to N for the DEFAULTS record, as shown in Figure 10-3 on page 235.

```

CIU250          CICS Interdependency Analyzer for z/OS - V2R2M0      2008/04/23
                DB2/MQ/IMS Resource Options for                    09:54:58AM

                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          MQ Options          IMS Options
DB2 . . . . . N      MQ . . . . . N      IMS . . . . . N

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC46

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

```

Figure 10-3 CINT: Set MQ option to 'N' for the defaults.

4. Set MQ collection on for our CICS region, IYDZZ328. Return to the Region Configuration Menu, and select option 5 against the IYDZZ328 region, as shown in Figure 10-4 on page 236.

```

CIU200          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/23
                  Region Configuration Menu                               10:12:52AM

Type action code then press ENTER.                                     More :
1=Add Region      3=Delete Region   5=DB2/IMS/MQ Options  7=Affinity Options
2=Copy Region    4=CICS Options     6=General Options   8=Time/Date Options

      CICS      CICS  New      New
Act  Applid   Sysid  Applid   Sysid  Status      Collecting
DEFAULTS  DFTS
ALL      ALL
IYDZZ138  Z138
IYDZZ228  Z228
IYDZZ238  Z238
IYDZZ318  Z318
5  IYDZZ328  Z328
                                     UNCONNECTED
                                     UNCONNECTED
                                     UNCONNECTED
                                     UNCONNECTED
                                     STOPPED

CICS Sysid:  Z328   CICS Applid:  IYDZZ328   TermID:  TC46

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

```

Figure 10-4 CINT: CHange MQ option for region IYDZZ328

- In the DB2/MQ/IMS resource option menu, set the MQ option to Y for the IYDZZ328 region, as shown in Figure 10-5 on page 237.

```

CIU250          CICS Interdependency Analyzer for z/OS - V2R2M0      2008/04/23
                DB2/MQ/IMS Resource Options for                    10:30:52AM

                CICS Sysid   : Z328      CICS Applid   : IYDZZ328

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

Detect command types: Y=Yes, N=No or blank=default

DB2 Options          MQ Options          IMS Options
DB2 . . . . .      MQ . . . . . Y      IMS . . . . .

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC46

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

```

Figure 10-5 CINT: Collect MQ information for region IYDZZ328.

You can control the collection of DB2, MQ and IMS from option 2 in the CINT transaction, for example, you might only have MQ in one of your CICS regions, so you then want to switch off MQ in all other regions.

Example: Specifying CICS resources to collect

The default for CICS resources is to collect all resources. From the CICS Options panel, you can select the CICS resources that you want to collect dependency information for. To look at the defaults, select option 4 against the DEFAULTS entry in the Region Configuration menu, as shown in Figure 10-6 on page 238 is shown.

From this menu, you can decide which CICS resource types you want to collect. You might not be interested in collecting SPI information, so you could change all of these options to N. If you want to collect different resources in different regions, you can override the defaults for each region using the same technique that you used for the MQ override option.

```

CIU240          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/23
                  CICS Resources Options for                          02:18:46PM
                  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 . . . . . Y Others . . . . . Y

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 TCIPService . Y FEPI . . . . . Y Journals . . . Y
Library . . . Y IPCONN . . . . Y

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC46

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

```

Figure 10-6 CINT: DEFAULT CICS Options menu

For the following resources, you can select to collect detailed information:

- ▶ Program
- ▶ Transaction
- ▶ TSQueue
- ▶ TDQueue
- ▶ File
- ▶ Webservices

To collect detail information, select D against these resource options. Because you selected D, CICS IA collects the usual EXEC CICS command usage information into the VSAM files. It also collects detailed information for these resources and stores the data in a VSAM file.

Why collect detail information

Detailed information for these resources can assist you in resource deployment from development to test and subsequently from test to production. It also assists with threadsafe analysis because we have more information about the resource, for example, we know the file type. Is it RLS or local?

The detailed information that we collect is the same information that you get back from an INQUIRE resource command.

Note: Collecting detailed information increases the path length through the exits. Only collect detailed information if required.

The detailed information is shown in the new CICS IA Explorer. For more information, see Chapter 6, “Analyzing CICS IA data using the CICS IA Explorer” on page 121.

10.1.2 Excluding programs or transactions from the collector

CICS IA gives you the ability to exclude programs or transactions from the collection. It is shipped with default programs called CIUXPROG and CIUXTRAN. The source for these is in the SCIUSRCE dataset. The CIUXTRAN is a dummy part and contains no transactions. The CIUXPROG contains program prefixes for IBM programs, as shown in Figure 10-7 on page 240.

```

CIUXPROG CSECT
CIUXPROG AMODE 31
CIUXPROG RMODE ANY
        DS      OF
* 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(4),C'DSNC'       DB2
      DC      AL1(4),C'DSN2'       DB2
      DC      AL1(3),C'EYU'        CICSPlex SM
      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(4),C'IN25'
      DC      AL1(0)                End of list
END      CIUXPROG

```

Figure 10-7 IBM supplied program exclude list

Example: Creating an exclude list

You can create your own program or transaction to create exclude lists for your environment. You can also create different exclude lists for different regions. In the this example, we exclude all transactions that start with a prefix of AFT from the collector for CICS region IYDZZ328.

To create an exclude list:

1. First, copy the supplied source for CIUXTRAN from SCIUSRCE into your own library, and call it CIUXT328.
2. Add in the transaction prefix, as shown in Figure 10-8 on page 241.

```

EJECT
CIUXTRAN CSECT
CIUXTRAN AMODE 31
CIUXTRAN RMODE ANY
        DS    OF
* Add user prefixes here
*        DC    AL1(3),C'TST'           Example
        DC    AL1(3),C'AFT'           Exclude AFT transactions
        DC    AL1(0)                   End of list
        END  CIUXTRAN

```

Figure 10-8 Add transaction prefix AFT to exclude list CIUXT328

- Use the supplied sample JCL, CIUJCLXT in SCIUSAMP, to assemble and link this code, as shown in Figure 10-9. Make sure that you link it into a load module that is in the DFHRPL for the selected CICS region or regions.

```

//ASM      EXEC PGM=ASMA90,
//          PARM='OBJECT,LIST,ALIGN,TERM'
//SYSIN    DD DSN=CICSIAD.CICSIAD.ASM(CIUXT328),
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTEM   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 DSN=CICSIAD.IYDZZ328.LOADLIB(CIUXT328),
//          DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=&&OBJ,DISP=(OLD,DELETE)
//*-----*

```

Figure 10-9 CIUJCLXT to assemble and link CIUXT328

- After you create this load module, tell CICS IA to use this exclude list for the CICS region IYDZZ328. Return to the Region Configuration Menu, and select option 6 against the IYDZZ328 region.

5. Enter CIUXT328 for the transaction exclude list, as shown in Figure 10-10.

```
CIU260          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/24
                  General Options for                                     09:15:01AM
                  CICS Sysid   : Z328   CICS Applid   : IYDZZ328

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

Control options (Fields may also be set to blanks for default)
Data to Collect . . . . . : (Y=Affinity, N=Interdependency)
Perform periodic saves . . . : (Y=Yes, N=No)
Trigger for CINB start . . . : (2 to 9999 thousand record updates)
Inquire for DB2 resources. . . : (Y=Yes, N=No)
Restore data on start . . . : (Y=Yes, N=No)
Multiple signon with same id : (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 . . : CIUXT328 (1 to 8 characters)

CICS Sysid: Z328   CICS Applid: IYDZZ328   TermID: TC68

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

Figure 10-10 Define exclude list CIUXT328 for region IYDZZ328.

10.1.3 Using the Timer options

CICS IA gives you the ability to control at what time of day you want to collect affinity or Dependency resource information. You can set these options for all regions using the DEFAULT entry or for each individual CICS region. In this case, the DEFAULT region is set up to collect resource information at all times, as shown in Figure 10-11 on page 243.

```

CIU280          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/24
                  Time and Date Options for                          09:56:35AM
                  CICS Sysid: DFTS      CICS Applid: DEFAULTS

Modify the options and press Enter to update or F12 to Cancel.

Time and date 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 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

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: Z328      CICS Applid: IYDZZ328      TermID: TC68

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

```

Figure 10-11 DEFAULT Time and Date Options - on all the time

Example: Set new time and date options for region IYDZZ328

For CICS region IYDZZ328, we set the time and date options to meet the following criteria:

- ▶ Hour of the day: 9am to 5pm
- ▶ Day of the week: Monday through Friday
- ▶ Day of the month: Every 3rd day to reduce data
- ▶ Month of the year: Avoid peak time in December

To set new time and date options for region IYDZZ328:

1. Return to the Region Configuration Menu, and select option 8 against the IYDZZ328 region.
2. Enter the options, as shown in Figure 10-12 on page 244.

Note: For the times that you do not want CICS IA to collect, you must enter N to override the DEFAULT values. If you leave it blank, the default value is used, which in this case is Y.

```

CIU280          CICS Interdependency Analyzer for z/OS - V2R2M0          2008/04/24
                  Time and Date Options for                            10:09:54AM
                  CICS Sysid: Z328    CICS Applid: IYDZZ328

Modify the options and press Enter to update or F12 to Cancel.

Time and date slots: Y=Yes, N=No or blank=default

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 N N N N N N N N Y Y Y Y Y Y Y Y N N N N N N N N

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

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 N N Y N N Y N N Y N N Y N N Y N N Y N N Y N N Y N N Y N N N 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   N

CICS Sysid: Z328    CICS Applid: IYDZZ328    TermID: TC68
CIU2111I CINT options are updated
F1=Help    F2=          F3=Exit    F4=          F5=          F6=
F7=        F8=          F9=          F10=         F11=         F12=Cancel

```

Figure 10-12 Time and date options for region IYDZZ328.

Note: In this example, we chose not to collect data in December because of performance concerns. By doing this, we could miss the collection of resource information for transactions and programs that only run during year-end processing in December.

10.1.4 Other General Options that can affect performance

In the General Options menu, shown in Figure 10-10 on page 242, there are other options that can affect performance:

- ▶ Perform periodic saves
- ▶ Trigger for CINB
- ▶ Inquire on DB2 resource
- ▶ Restore data on start
- ▶ Maintain usage counts

We discuss ‘Inquire on DB2 resources’ in the next section, 10.2.1, “Database load performance” on page 248.

The other four options are related. We start by describing what they are and then describe how they are related:

- ▶ Perform periodic saves

If this option is set to Y, the Collector saves data from the data space to the VSAM files every five minutes. It does this by waking the CINB long running task.

- ▶ Trigger for CINB

If this option is set to 2 or greater, it tells the Collector to save data from the data space to the VSAM file after 2000 changes or 'n' thousand changes, where 'n' is the value you selected, are made to the data space. If a value of 1 is selected, then this trigger is switched off.

What do we mean by a *change*? The control increments a change count when an entry is added to the data space the very first time or when an entry in the data space is flagged as CHANGED. See the description of Maintain usage counts:

- ▶ Restore data on start

If this option is set to Y, the Collector loads the data space with information from the VSAM files for that CICS region. If the option is set to N, the Collector will not reload the data space, and it will also delete the VSAM records that are associated with that CICS region.

- ▶ Maintain usage counts

If this option is set to Y, the Collector updates the usage count, and the last used timestamp is set for that EXEC CICS entry in the data space. If this option is set to N, the EXEC CICS entry is never updated. We only add it to the data space the first time we see it.

The usage count and last used timestamp are updated in the data space every time an entry is seen. The usage count is used to flag whether the entry should be written to the VSAM file, which is done by a CHANGE flag:

- ▶ If the usage count is less than or equal to 10, the flag gets set.
- ▶ If the usage count is less than or equal to a 1000, but greater than 10, the flag gets set every 10th update.
- ▶ If the usage count is greater than 1000, the flag gets set every 100th update.
- ▶ If the CHANGE flag is set after an entry update, the CHANGE count is increased by one.

Using the above options, you have many ways to control how often the background transaction CINB runs in order to write from the data space to VSAM.

Here are some recommendations:

If you are in a stable environment and do not require the use of the usage count and the last used timestamp, set the options as follows:

- ▶ Perform periodic saves = Y, save every 5 minutes.
- ▶ Trigger for CINB = 1, do not use the trigger feature.
- ▶ Restore data on start = Y, restore existing entries.
- ▶ Maintain usage counts = N, only add a new entry

In this case after you have been running for a while the number of new entries being added should decrease since you have already captured the information for nearly all the EXEC CICS calls. Reloading the data at collector start also helps as you will only read an entry from the data space each time you see it.

If you are in a changing environment such as development and again do not require usage counts then set the options as follows:

- ▶ Perform periodic saves = Y, save every five minutes.
- ▶ Trigger for CINB = 1, do not use the trigger feature.
- ▶ Restore data on start = N, do not restore existing entries, and delete the VSAM records that are associated with this CICS region.
- ▶ Maintain usage counts = N, only add a new entry.

For either of the scenarios, you could choose to switch periodic saves off and not have the CINB transaction run at all. CINB would run when the Collector is stopped by operations or at CICS shutdown. The danger here is that you could lose the data you collected if an outage occurs in the region.

If you are in an environment where you need to maintain the usage counts and the last used time stamp, you are likely to see a number of updates to the data space and subsequently to the VSAM file. To prevent CINB from running continuously, set the options as follows:

- ▶ Perform periodic saves = N
- ▶ Trigger for CINB = 50, save to VSAM every 50,000 changes
- ▶ Restore data on start = Y
- ▶ Maintain usage counts = Y

If this was a changing environment, set 'restore data on start' to N. You could also only save to VSAM when CICS IA stops by setting the 'Trigger for CINB' to 1.

Cloned systems

Many users clone CICS regions across their CICSplexes, so many regions use exactly the same set of applications and resources. It is pointless to turn on

collection in all of these regions because that would just result in a Dependency database with duplicated data that is differentiated only by region.

Therefore, in the case of cloned regions, we suggest that you turn on collection in a subset of those regions, maybe two or three, to collect the necessary data to cover all of the cloned regions.

10.2 Transaction CINB performance

When you collect DB2 resources, the Collector, when writing from the data space to the VSAM file, uses two IBM DB2 tables to obtain the resource name. These tables are: SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT.

In many environments, these tables are extremely large and under managed. Initially users were having performance problems with the CINB transaction when fetching data from these tables. CICS IA introduced some indexes to assist with the problem. These indexes are in the SCIUSQL dataset in member CIUIBM1. This resolved the performance issue for many users. Other users had their DB2 system programmers create alternative indexes based on their environments. The current recommendation is to create the alternative indexes, as shown in Example 10-1.

Example 10-1 DB2 Indexes for SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT.

```
CREATE INDEX SYSPKID1
  ON SYSIBM.SYSPACKSTMT
  ( "NAME"          ASC ,
    "STMTNO"        ASC ,
    "SECTNO"        ASC ,
    "SEQNO"         ASC )
  USING STOGROUP SYSDEFLT
    PRIQTY 2880
    SECQTY 720
  ERASE NO
  FREEPAGE 0
  PCTFREE 20
  GBPCACHE CHANGED
  BUFFERPOOL _db2bfpi_
  CLOSE NO
  COPY NO;
```

```
CREATE INDEX SYSSMTID1
  ON SYSIBM.SYSSTMT
  ( "NAME"          ASC ,
```

```

"STMTNO"      ASC  ,
"SECTNO"      ASC  ,
"SEQNO"       ASC  )
USING STOGROUP SYSDEFLT
              PRIQTY 2880
              SECQTY 720
              ERASE NO
FREEPAGE      0
PCTFREE 20
GBPCACHE CHANGED
BUFFERPOOL _db2bfpi_
CLOSE NO
COPY NO;

```

If you are still having problems with collecting DB2 resource information, consider one of the following:

- ▶ Collect only DB2 resource information. Do not collect any CICS, WMQ, or IMS resource information simultaneously.
- ▶ Limit your collection by adding unwanted programs and transactions to an exclude list.
- ▶ Only save to VSAM when the Collector is stopped.
- ▶ Do not inquire on the SYSIBM tables when you save to VSAM. To do this, set the 'Inquire on DB2 resource' option to N.

Note: If CINB is inquiring on SYSIBM tables, the user ID under which CINB is running needs access to the SYSIBM tables, which you can do by giving the user ID access to the CICS plan that the CIUCINB2 program is bound in. See sample JCL CIUDBND in SCIUSAMP.

If you have not inquired on the SYSIBM tables, you can use the V_CIU_DB2_RES or V_CIU_DB2_RES2 to view the actual DB2 statements that the program uses. To do this, the DB2 resource information table must be stored in a CICS IA table CIU_DB2_DATA in the same DB2 subsystem that it was collected.

10.2.1 Database load performance

In previous releases of CICS IA, there were fixes provided to improve the performance of the CICS IA database load jobs. Ensure that you are up-to-date with CICS IA PTF service levels.

In the previous releases of CICS IA, we created tables and views to query indirect resources that the programs used. These tables and views were:

- ▶ CIU_CICS_CHAINP
- ▶ CIU_CICS_CONN
- ▶ CIU_CICS_CHAINP_T
- ▶ V_CIU_CICS_INDS2

They are no longer used and of no real value. They are updated by step STEP051 in sample jobs CIUUPDB and CIUUPDB1. They have already been commented out in CICS IA V2.2 and will be removed in the next release of CICS IA. If you are running this step, we recommend that you remove it.

10.2.2 Database query performance

Both the DB2 load jobs and the DB2 query interfaces can get improved performance by running the DB2 utilities.

For DB2 V8 and later, these utilities are shipped with the DB2 Utilities Suite for z/OS, V8 (5655-K61), or later.

Using DB2 runstats

After loading CICS IA with a large amount of data, we recommend that you run the DB2 reorg and runstats utilities. You must then rebind all the CICS IA packages.

Run a reorg on all of the CICS IA table spaces, as shown in Example 10-2.

After you run this job, you need to rebind by running the sample job, CIUDBND, which is in the dataset SCIUSAMP.

Example 10-2 RUNSTATS job for CICS IA table spaces

```
//CIUREORG JOB USER=JAMESE,NOTIFY=JAMESE,  
//          CLASS=A,MSGCLASS=Y,REGION=OM  
//*  
//DB2LIB JCLLIB ORDER=DSN8102F.PROCLIB  
//STEP010 EXEC DSNUPROC,  
//          SYSTEM=DG2F,  
//          UID='CIU',  
//          UTPROC=''  
//STEPLIB DD DSN=SYS2.DB2.V810.SDSNLOAD,  
//          DISP=SHR  
//* CIUAPNON IS ALWAYS REQUIRED  
//DSNUPROC.SYSREC DD DUMMY  
//DSNUPROC.SYSIN DD *
```

```

REORG TABLESPACE CICSIA3.CIUVER1 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC1 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC2 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC3 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC4 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC5 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC6 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC8 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCIC9 NOSYSREC
REORG TABLESPACE CICSIA3.CIUCICA NOSYSREC
REORG TABLESPACE CICSIA3.CIUSB2D NOSYSREC
REORG TABLESPACE CICSIA3.CIUFILE NOSYSREC
REORG TABLESPACE CICSIA3.CIUPROG NOSYSREC
REORG TABLESPACE CICSIA3.CIUTRANS NOSYSREC
REORG TABLESPACE CICSIA3.CIUTDQ NOSYSREC
REORG TABLESPACE CICSIA3.CIUEXIT NOSYSREC
REORG TABLESPACE CICSIA3.CIUMQD NOSYSREC
REORG TABLESPACE CICSIA3.CIUIMSD NOSYSREC
REORG TABLESPACE CICSIA3.CIUCSSD NOSYSREC
REORG TABLESPACE CICSIA3.CIUCSSE NOSYSREC
REORG TABLESPACE CICSIA3.CIUREGD NOSYSREC
REORG TABLESPACE CICSIA3.CIUERSRC NOSYSREC
REORG TABLESPACE CICSIA3.CIUWEBS NOSYSREC
REORG TABLESPACE CICSIA3.CIUTSQ NOSYSREC
REORG TABLESPACE CICSIA3.CIUTSCTS NOSYSREC
REORG TABLESPACE CICSIA3.CIUAFFD NOSYSREC
REORG TABLESPACE CICSIA3.CIUVER1 NOSYSREC

```

/*

You need to run a full runstats (index all, column all) on all of the CICS IA tables.

After running this job, you need to rebind by running the sample job, CIUSBND, which is in the dataset SCIUSAMP.

10.3 IA database cleanup

CICS IA data that you collect from development or test can change, often leading to redundant records in the database. There are a number of ways to purge data from the database. In this section, we look at SQL queries to help you purge unwanted data from your database tables, which improves performance.

Note: Remember that the CICS IA data is loaded in the DB2 tables from the VSAM files. You must make sure that you clear the VSAM file and the DB2 tables. For development and test regions, we recommend that the files are emptied at CICS IA startup.

CICS IA supplies a number of sample SQL jobs to purge data from the CICS IA tables. These jobs are in SCIUSQL and start with the prefix CIUP, for example, CIUPCICS deletes data from the CIU_CICS_DATA. The sample jobs use the LAST_RUN date. The query first displays the records before the selected LAST_RUN date, as shown in Figure 10-13.

```
--  
-- Show rows that are older then specified timestamp  
SELECT APPLID, TRANSID, PROGRAM, FUNCTION, TYPE, OBJECT  
  FROM CIU_CICS_DATA READONLY  
  WHERE LAST_RUN<='2008-01-01-00.00.00.000000';  
-- Uncomment this statement when you want to delete rows  
--DELETE FROM CIU_CICS_DATA  
--      WHERE LAST_RUN<='2008-01-01-00.00.00.000000';  
COMMIT;
```

Figure 10-13 Sample query to delete all records before 2008.

Another method is to use delete by program version, which you can do by querying on the program, the program length, and the first used timestamp. Program and program length are part of the unique key that is used when storing data in the four main tables:

- ▶ CIU_CICS_DATA
- ▶ CIU_DB2_DATA
- ▶ CIU_MQ_DATA
- ▶ CIU_IMS_DATA

When a program changes, the program length probably changes too. CICS IA uses this as a crude method for a programming version. The EXEC commands for a changed program contain a FIRST_RUN timestamp, which is used to identify the latest program. We can then delete all of the commands that are reported for previous programs. Now, we look at how this command is created.

First, we examine the data to look for programs with different program lengths. We can see from Figure 10-14 on page 252 that program DB900001 has two different program lengths. From the FIRST_RUN timestamp, we can see that program length 2180 is the latest program.

We now need to get the program length for the latest programs. The query in Figure 10-16 shows the program and program length for the latest used programs. Again, we concatenate the output for use in a SQL sub query.

```

SELECT DISTINCT APPLID||PROGRAM||PROGLEN
FROM CIU_CICS_DATA , (
  SELECT DISTINCT APPLID AS A, PROGRAM AS P,
    MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
  FROM CIU_CICS_DATA
  WHERE APPLID='IYCYZC37'
  GROUP BY APPLID, PROGRAM) X
WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S;
-----+-----+-----+-----+-----
-----+-----+-----+-----+-----
IYCYZC37DB90000100002180
IYCYZC37DB910001000032A0
IYCYZC37DB93000100002000
IYCYZC37DB94000100002698
IYCYZC37EMSTESTA00002110
IYCYZC37EMSTESTB00001500
IYCYZC37EMSTESTS00003380
IYCYZC37FC01000100000AD0
IYCYZC37FC02000100001558

```

Figure 10-16 Show latest program and program lengths.

We can now display the records for all of the commands that were collected from earlier versions of the programs, which we show in the query in Figure 10-17 on page 254. We can now choose to delete these records, as shown in Figure 10-18 on page 254.

Now, when we run the first query again, we can see that there is only one entry for program DB900001, as shown in Figure 10-19 on page 255.

You can use these queries as the basis for writing your own delete functions.

```

SELECT * FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
  AND APPLID||PROGRAM||PROGLEN NOT IN
(SELECT DISTINCT APPLID||PROGRAM||PROGLEN
 FROM CIU_CICS_DATA , (
  SELECT DISTINCT APPLID AS A, PROGRAM AS P,
    MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
  FROM CIU_CICS_DATA
   WHERE APPLID='IYCYZC37'
   GROUP BY APPLID, PROGRAM) X
 WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S)
-----+-----+-----+-----+-----+-----+-----+
APPLID  HOMESYSID  TRANSID  PROGRAM  FUNCTION  TYPE      OBJECT
-----+-----+-----+-----+-----+-----+-----+
IYCYZC37 TS07      DB90     DB900001 RECEIVE  MAP       S1
IYCYZC37 TS07      DB90     DB900001 RECV MAP MAPSET    CBMS01
IYCYZC37 TS07      DB90     DB900001 SEND     MAP       S1
IYCYZC37 TS07      DB90     DB900001 SEND     MAP       S1
IYCYZC37 TS07      DB90     DB900001 SEND MAP MAPSET    CBMS01
IYCYZC37 TS07      DB90     DB900001 SEND MAP MAPSET    CBMS01
IYCYZC37 TS07      DB91     DB910001 RECEIVE  MAP       S2

```

Figure 10-17 Commands to be deleted

```

-----+-----+-----+-----+-----+-----+-----+
DELETE FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
  AND APPLID||PROGRAM||PROGLEN NOT IN
(SELECT DISTINCT APPLID||PROGRAM||PROGLEN
 FROM CIU_CICS_DATA , (
  SELECT DISTINCT APPLID AS A, PROGRAM AS P,
    MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
  FROM CIU_CICS_DATA
   WHERE APPLID='IYCYZC37'
   GROUP BY APPLID, PROGRAM) X
 WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S)
-----+-----+-----+-----+-----+-----+-----+
DSNE615I NUMBER OF ROWS AFFECTED IS 24
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+-----+-----+

```

Figure 10-18 Delete commands for old programs


```

SELECT DISTINCT APPLID, PROGRAM, PROGLEN,
MAX(CAST(FIRST_RUN AS CHAR(26)))
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM , PROGLEN;
-----+-----+-----+-----+-----+-----
APPLID    PROGRAM    PROGLEN
-----+-----+-----+-----+-----+-----
IYCYZC37  DB900001   00002180  2007-11-22-11.23.57.000000
IYCYZC37  DB910001   000032A0  2007-11-22-11.23.57.000000
IYCYZC37  DB930001   00002000  2007-11-22-11.23.57.000000
IYCYZC37  DB940001   00002698  2007-11-22-11.23.57.000000
IYCYZC37  EMSTESTA   00002110  2008-04-03-12.38.48.000000

```

Figure 10-19 Updated program version query.



Debugging

In this chapter, we begin by discussing how to debug problems that you might find during installation and customization of the product. We address how to debug some of the major functional areas, and we cover some debugging facilities.

11.1 Installation and customization

We cover CICS IA configuration in Chapter 2, “Installation and customization” on page 11. In this section, we review things that could go wrong.

11.1.1 What to look for if something goes wrong

CIUDBCR

It is important to involve the DB2 administrator to review all of the parameters that are involved in this job before you run it. If the job fails with SQLCODE -904 and REASON 00D70025, this indicates that there is a shortage of space on the DASD volume or volumes that are specified in the member CIUMAIN in library SCIUSQL.

CIUDBND

If you do not run this job during customization or after any of the load modules are replaced by service, SQLCODE -905 will result. If you get this return code, the first course of action is to rerun this job.

CIUALOAD

If you have not run this, a CINC query fails with CIU7017I No SQL available for this query. If you see this message, the first course of action is to review the application definition jobs, and rerun at least CIUALOAD.

11.2 The Collector component

We discuss the Collector component in Chapter 4, “The Collector” on page 73. In this section, we review things that could go wrong with the Collector.

11.2.1 What exits are invoked

CICS provides Global User Exit points that enable you to write exit programs to perform various tasks when ever the exit is driven. An example of this is the exit point XEIOUT, which is driven after an application program issues any EXEC CICS API or SPI command. CICS IA uses this capability to inspect the CICS command and to extract relevant Dependency data. This data is then recorded in a data space before application execution continues. CICS IA uses a data space so that the performance impact of recording this data at the exit points is minimized.

Dependency data is collected by CICS global user exit programs at the following exit points:

- ▶ XFCREQC and XFCSREQC for File Control
- ▶ XICREQC for Interval Control
- ▶ XPCREQC and XPCFTCH for EXEC CICS LINK
- ▶ XTDEREQC for Transient Data
- ▶ XTSEREQC for Temporary Storage
- ▶ XEIIIN & XEIOOUT for all other CICS commands
- ▶ XRMIOUT for MQ, DB2 EXEC SQL, and IMS EXEC DLI
- ▶ XRMIIN for MQ

The Collector status is also written to the CINT log (an Extrapartition TD Queue) together with the status of the user exit programs. Every change of status is reflected by CICS IA in this log, so it is possible to see when the Collector was started, paused, continued, or stopped. The save transaction, CINB, also writes status records to the log that indicates when and why it was started, when it completed, and how many records it saved to the VSAM data sets, as shown in Figure 11-1.

```
15:51:42 IYDZZ318 CINT being used by CICSUSER
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXDUMM, EXIT XICEXP OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCC01, EXIT XEIOOUT OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCC01, EXIT XEIIIN OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCF1, EXIT XFCREQC OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCF2, EXIT XFCSREQC OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCP2, EXIT XPCFTCH OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCP1, EXIT XPCREQC OK
15:51:49 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCS1, EXIT XTSEREQC OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCD1, EXIT XTDEREQC OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCC11, EXIT XICEREQC OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT for program CIUXCCR1, EXIT XRMIOUT OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCC01 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCF1 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCF2 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCP2 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCP1 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCS1 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCD1 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCC11 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCR1 OK
15:51:50 IYDZZ318 EXEC CICS ENABLE EXIT START for program CIUXCCR1 OK
15:51:50 IYDZZ318 Interdependency collector is now RUNNING
15:51:50 IYDZZ318 Dependency files are emptied
15:51:50 IYDZZ318 CINB task is starting
15:51:52 IYDZZ318 CINT session has ended
15:56:50 IYDZZ318 CINB save started - because of TIME
15:56:50 IYDZZ318 CINB save ended - 85 records saved
15:56:50 IYDZZ318 CINB save started - because of STOP
15:56:50 IYDZZ318 CINB save ended - 0 records saved
```

Figure 11-1 Status information written to the CINT log

In the Configure Region Options menu, you can choose which CICS, DB2, MQ, and IMS commands are collected.

11.2.2 What to look for if Collection is not happening

If data is not being collected, consider the following:

- ▶ Make sure that the CICS IA exit programs are enabled after any other exit programs at the same exit point. To ensure this, run the CINT transaction only after the other exit programs are enabled.
- ▶ Check that the Collector is running by looking for its status in the CINT Operations menu or in the CINT Extrapartition TD Queue. (This is available in the SDSF job output for the CICS region if the //CINT DD card is assigned to SYSOUT=*.) If it is in the STOPPED or PAUSED state, no collection occurs. See Figure 11-1 on page 259.
- ▶ Check that the relevant command types were selected in the CICS, DB2, IMS, and MQ options in the CINT Region Configuration menu.
- ▶ Check that the specification of the Transid prefix in the CINT General Options menu is correct. If a prefix is specified, Dependency data is collected only for transaction IDs that start with that prefix.
- ▶ Check that the Collector was stopped and then started after changing any region-specific options, for example, if you start the Collector with the CICS options all set to N, change some of the options to Y. No collection for these options occurs until the Collector is stopped and restarted. You can check for this in the CINT log by examining whether the exits were ENABLED and STARTed. (See Figure 11-1 on page 259). Figure 11-2 shows an example of the Collector being started with no options switched on.
- ▶ For DB2 collection, access to the SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT table might be required. Check the CINT log for SQL error codes. We cover common SQL error codes in 11.3.1, “Resolving DB2 SQL error codes” on page 261.

```
CIU2240I 09/20/04 09:43:00 SCSCPAA1 EXEC CICS ENABLE EXIT for program
CIUXDUMM, EXIT XICEXP OK
CIU2231I 09/20/04 09:43:01 SCSCPAA1 Number of records restored = 0
CIU2214I 09/20/04 09:43:01 SCSCPAA1 Collector is now RUNNING
CIU3301I 09/20/04 09:43:01 SCSCPAA1 CINB task is starting
```

Figure 11-2 CINT log showing the Collector starting with no options switched on

11.3 CINT transaction

In this section, we review problems that could occur when you use the CINT transaction. The main problems here are DB2 bind and access problems.

11.3.1 Resolving DB2 SQL error codes

To assist in resolving the following SQL error codes -922 and -923, check the following:

- ▶ Check that the correct DB2TRANSACTION definitions are defined in the CICS region, as shown in Figure 11-3.

```
I DB2TRAN
STATUS: RESULTS - OVERTYPE TO MODIFY
Db2t(CIA0 ) Db2e( CICSIAAD ) Tra( CIA0 ) Plan(CICSIA3P)
Db2t(CIA1 ) Db2e( CICSIAAD ) Tra( CIA1 ) Plan(CICSIA3P)
Db2t(CIA2 ) Db2e( CICSIAAD ) Tra( CIA2 ) Plan(CICSIA3P)
Db2t(CIA3 ) Db2e( CICSIAAD ) Tra( CIA3 ) Plan(CICSIA3P)
Db2t(CIA4 ) Db2e( CICSIAAD ) Tra( CIA4 ) Plan(CICSIA3P)
Db2t(CID0 ) Db2e( CICSIAAD ) Tra( CID0 ) Plan(CICSIA3P)
Db2t(CID1 ) Db2e( CICSIAAD ) Tra( CID1 ) Plan(CICSIA3P)
Db2t(CID2 ) Db2e( CICSIAAD ) Tra( CID2 ) Plan(CICSIA3P)
Db2t(CID3 ) Db2e( CICSIAAD ) Tra( CID3 ) Plan(CICSIA3P)
Db2t(CID4 ) Db2e( CICSIAAD ) Tra( CID4 ) Plan(CICSIA3P)
Db2t(CIH2 ) Db2e( CICSIAAD ) Tra( CIH2 ) Plan(CICSIA3P)
Db2t(CII0 ) Db2e( CICSIAAD ) Tra( CII0 ) Plan(CICSIA3P)
Db2t(CII1 ) Db2e( CICSIAAD ) Tra( CII1 ) Plan(CICSIA3P)
Db2t(CII2 ) Db2e( CICSIAAD ) Tra( CII2 ) Plan(CICSIA3P)
Db2t(CII3 ) Db2e( CICSIAAD ) Tra( CII3 ) Plan(CICSIA3P)
Db2t(CII4 ) Db2e( CICSIAAD ) Tra( CII4 ) Plan(CICSIA3P)
Db2t(CIM0 ) Db2e( CICSIAAD ) Tra( CIM0 ) Plan(CICSIA3P)
Db2t(CIM1 ) Db2e( CICSIAAD ) Tra( CIM1 ) Plan(CICSIA3P)
```

Figure 11-3 CEMT INQ on DB2TRAN

- ▶ Check that there is a CICS resourcer definition for the DB2ENTRY, as shown in Figure 11-4

```
I DB2ENTRY
STATUS: RESULTS - OVERTYPE TO MODIFY
Db2e(CICSIAAD ) Use Ena Poo Equ Pro( 0000 ) Pth(0000)
Thread1( 0015 ) Threads(0000) Tpo Plan( CICSIA3P )
```

Figure 11-4 CEMT INQ on DB2ENTRY

- ▶ Check that the user(s) was granted access to this PLAN. Check the output from the CIUDBND sample job and CIUGRANT in SCIUSQL. If you are using RACF groups, ensure that the user is in the group.

Another common SQL code that we saw while using the CINC transaction is the -805, which indicates a BIND error. In this case, service was probably applied and the sample job CIUDBND has not run.

11.4 The CICS IA Explorer

In this section, we review problems that could occur when using the CICS IA Explorer. We cover the installation and configuration of the Explorer in Chapter 2, “Installation and customization” on page 11.

11.4.1 What level of the CICS IA Explorer am I using

You can obtain what level of the CICS IA Explorer you are using by looking at the information window. From the Explorer, click **Help** → **About IAExplorer**. The window shown in Figure 11-5 opens.

This information about this window is the latest service level of CICS IA.

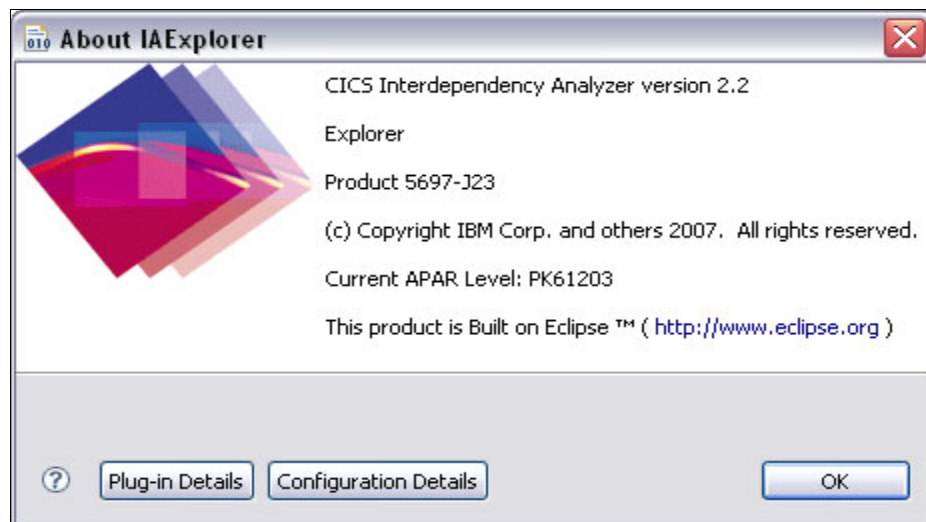


Figure 11-5 CICS IA Explorer information.

11.4.2 What if I get SQL error code -805

A -805 error code indicates that there is a BIND problem. However, we noticed that this code can be returned for other issues as well. APAR PK61203 resolved these instances. So, make sure that you have the latest CICS IA Explorer installed.

Another cause of the -805 error is because the incorrect plans were bound on to the z/OS DB2 database. You need to bind the following plans.

We bind the driver from UNIX® System Services using:

```
java com.ibm.db2.jcc.DB2Binder -url  
jdbc:db2://server_name:port_number/location_name -user  
YOUR_USERID_IN_CAPS -password YOUR_PASSWORD_IN_CAPS -action replace  
-size 20
```

For CICS users, we create a plan called DSNJCC, and bind the NULLID packages and so on into it.

11.5 CICS IA Trace Facility

For certain problem types, you might be asked to supply a CICS IA trace. Use the Global Options Menu to switch the trace facility on or off, as shown in Figure 11-6.

```
CIU300          CICS Interdependency Analyzer for z/OS - V1R3M0      2004/  
                  Global Options Menu                               10:42  
  
Modify the options and press Enter to update, or press PF12 to cancel.  
  
Control options  
  VSAM file sharing . . . . . : N (Yes/No)  
  High Level Trace . . . . . : Y (Yes/No)  
  
National Language Option . . . : E Code: ENU  
  
Date and Time Formats  
  Date . . . . . 4 1. MMDDYY 2. DDMMYY Separator . . . . . /  
                  3. YYMMDD 4. YYYYMMDD  
  
  Time . . . . . 1 1. 12 hrs 2. 24 hrs Separator . . . . . :
```

Figure 11-6 Global Options Menu showing the trace option

This trace is written to an internal trace table and is only visible from within a dump of the CICS IA address space, which we explain in the next section.

11.6 Taking a dump of CICS IA

For certain problem types, you might be asked to supply a dump of CICS IA. To request a dump of a CICS IA address space and data space at any time, use the MVS DUMP command.

Issue the MVS DUMP command from the console to dump a CICS IA address space and its associated data space. Use the ASID= keyword to identify the address space and the DSPNAME= keyword to identify the data space.

The data space name takes the form *nnnnn*INT where *nnnnn* is the five-character number that you can obtain from the CICS IA Operations Statistics Menu, which we show in Figure 11-7. It is usually 00000INT.

```
CIU150          CICS Interdependency Analyzer for z/OS - V1R3M0
                  Statistics Menu for

                CICS Sysid   : PAA1      CICS Applid   : SCSCPAA1

CINT state . . . . . : RUNNING
Number of pauses . . . . . : 0
Number of saves. . . . . : 0
Records written last save. : 0
Total records on file. . . : 222

Date/time of last start. . : 2004/09/08 10:53:58AM
Date/time of last save . . :
Date/time of last change . :

Total time RUNNING . . . . : 0000:00:05 (HHHH:MM:SS)
Total time PAUSED. . . . . : (HHHH:MM:SS)

Table dataspace name . . . : 00000INT   0.2 % full

CICS Sysid: PAA1   CICS Applid: SCSCPAA1   TermID: TC37
```

Figure 11-7 Statistics Menu showing the data space name

To obtain the ASID of the CICS address space where CICS IA is active and its associated data space name, use the MVS DISPLAY ACTIVE command:

```
/D A,SCSCPAA1
```

Figure 11-8 on page 265 shows output from the DISPLAY ACTIVE command.

```
RESPONSE=SC66
IEE115I 11.03.25 2004.252 ACTIVITY 348
  JOBS      M/S      TS USERS  SYSAS  INITS  ACTIVE/MAX VTAM  OAS
00005      00061    00008    00033  00062  00008/00050    00033
  SCSCPAA1 SCSCPAA1 CICS620  NSW  SO  A=008D  PER=NO  SMC=000
                                PGN=N/A  DMN=N/A  AFF=NONE
                                CT=069.552S  ET=18.10.43
                                WUID=STC28748  USERID=CICSTS
                                WKL=SYSTEM  SCL=SYSSTC  P=1
                                RGP=N/A      SRVR=YES  QSC=NO
                                ADDR SPACE  ASTE=02D0D340
                                DSPNAME=00000INT  ASTE=03F4E780
```

Figure 11-8 Output from the DISPLAY ACTIVE command

From the example in Figure 11-8, you can ascertain that the ASID is 008D and that the data space name is 00000INT. The data space name is also available in Figure 11-7 on page 264.

To dump this data space, use the following command, where *nn* and *mm* are the outstanding reply numbers:

```
/DUMP COMM=(IA DUMP)
/R nn,JOBNAME=SCSCPAA1,CONT
/R mm,DSPNAME=('SCSCPAA1'.00000INT),END
```

To format the dump, review the CICS IA User Guide.

11.7 The plan table

EXPLAIN is a monitoring tool that produces information about a plan, package, or SQL statement when it is bound. The output appears in a DB2 table called PLAN_TABLE, which is also called a *plan table*. Experienced DB2 users can use this table to give optimization hints to DB2. See the chapter about giving optimization hints to DB2 in the DB2 administration guides.

For certain problem types, you might be asked to provide the plan table to service personnel to assist with problem determination, analysis, and resolution. To record the monitoring tool output, bind the relevant programs with the EXPLAIN(YES) option, as shown in Figure 11-9 on page 266.

```

BIND PACKAGE (CPS3) -
MEMBER (CIUU050) -
OWNER (_own_) -
QUALIFIER (_qual_) -
LIBRARY ('_hlq_.SCIUDBRM') -
SQLERROR (NOPACKAGE) -
VALIDATE (BIND) -
ISOLATION (CS) -
EXPLAIN (YES) -
ACTION (REPLACE)

```

Figure 11-9 The BIND statement for CIUU050 showing the EXPLAIN option

PLAN_TABLE must already exist; otherwise, the bind fails with DSNX200I BIND SQL ERROR with TOKENS=owner.PLAN_TABLE. To create a plan table, refer to the *DB2 UDB for OS/390 and z/OS Administration Guide*, SC26-9931-03 for Version 7 or SC18-7413 for Version 8. The methods differ between the versions of DB2 being used.

After you create a plan table, recreate the problem, unload the plan table, and send the plan table to the requesting service personnel.



A

WebSphere Studio Asset Analyzer

So what, exactly, is WebSphere Studio Asset Analyzer (WSAA)?

WSAA 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 WSAA, you must first identify the production resources at your site. You then let WSAA 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 J2EE™ 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 Rational® ClearCase®.

The distributed scanners (crawlers) for non-z/OS resources run on either Microsoft Windows 2000 or Windows NT.

After WSAA 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 a WebSphere application, WSAA 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 WSAA 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.

WebSphere Studio Asset Analyzer helps your application development organization to:

- ▶ 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.

WSAA can be useful to a wide variety of groups in your organization that support all of the phases of the system development life cycle (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 WSAA in any phase of the application development process, including:

- ▶ Requirements
- ▶ Development
- ▶ Test
- ▶ Deployment

Important: WebSphere Studio Asset Analyzer requires a number of other program products to support it.

For organizations that are seeking to expand their existing applications to the Web, WSAA 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 WSAA enables them to drill down to understand the details of their application programs.

WSAA core functions

Figure A-1 is a depiction of the ways that you can use WSAA functions.

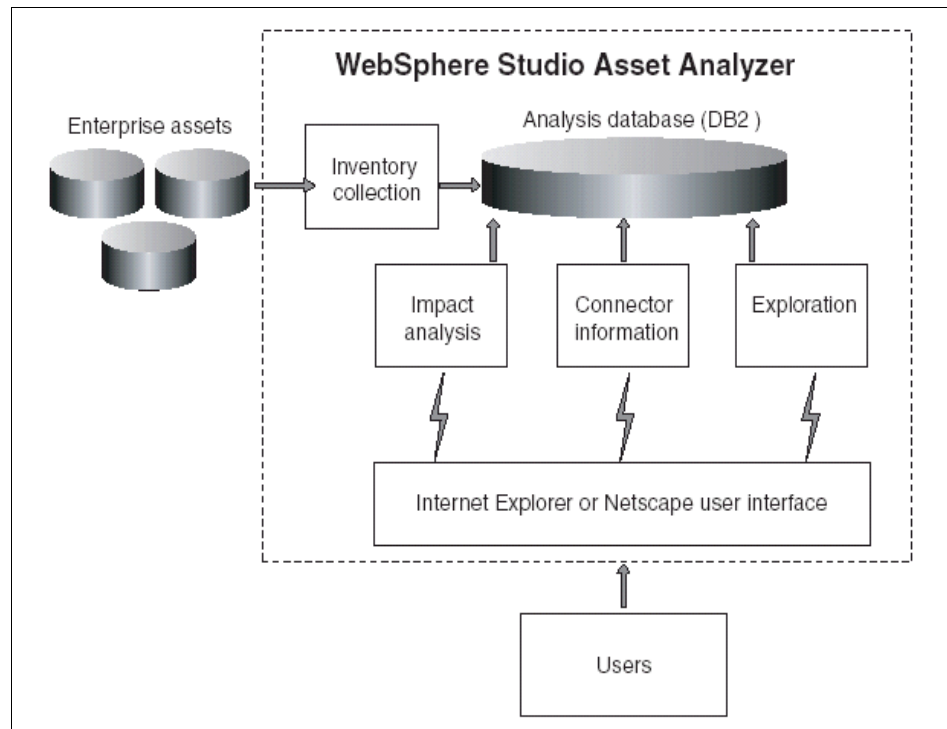


Figure A-1 An overview of WebSphere Studio Asset Analyzer

There are four core functions in WSAA, all of which you can access using a Web browser:

- ▶ Inventory collection
- ▶ Application exploration, including change impact analysis
- ▶ Connector building
- ▶ Help

We briefly describe each of these core WSAA functions in the following sections.

Inventory collection

Inventory collection forms the foundation of WSAA. You identify all of the production application resources at your site. You then instruct WSAA to process your site's application code and load it.

During inventory collection, WSAA's parsers scan the resources that you specify. The information is then stored in a DB2 database. WSAA collects inventory information for both mainframe and distributed applications.

Mainframe

Inventory is collected for OS/390 or z/OS components from:

- ▶ Source code

This code can be obtained from:

- Partitioned data sets (PDS or PDSE)
- IBM Source Control Library Manager (SCLM)
- Serena Software Inc.'s ChangeMan

The types of source code that can be scanned include:

- COBOL, PL/I, and Assembler, including copybooks and macros
- JCL, procs, and control cards

- ▶ CICS and IMS online regions, transactions, and subsystems

The mainframe scanners execute as batch jobs, which are submitted from either a Web browser or an ISPF session.

Distributed

Inventory is collected for Web-based components from:

- ▶ File systems, including the hierarchical file system (HFS) of UNIX System Services
- ▶ Any accessible WebDAV server
- ▶ Rational ClearCase

The distributed scanners are crawlers that run on Windows workstations.

Scanning results

As components are scanned into the database, WSAA establishes the appropriate logical connections among them. In this manner, copybooks are related to the programs that use them. In addition, programs are related to the batch jobs or online transactions that invoke them. Ultimately, application data files are linked to both.

Application exploration

Use the Explore function to view application components and their relationships after they are loaded into the database. When you explore your application, you can start at a very high level to obtain an overview of the general characteristics of your application. You can also drill down into the details of specific components, all the way down to the data element (that is, WORKING-STORAGE field) level.

You can search for components by looking for a specific one or by selecting one from a list of components. If you need to, you can search for components by:

- ▶ Name
- ▶ Application name
- ▶ Project name
- ▶ Site

We describe what some of these phrases mean in “Terminology” on page 272.

When you find the component that interests you, you can follow other component links to navigate through an application. Along the way, you can discover the answers to questions, such as:

- ▶ What program does a batch job or CICS transaction invoke?
- ▶ What subroutines does a particular program call?
- ▶ What files are used in a particular batch job step?

A key aspect of WSAA's Explore function is your ability to perform a change impact analysis. You can determine what components are affected based on changes to:

- ▶ Field declarations
- ▶ A section of program source code
- ▶ An entry point signature (for example, either a name or parameters)

You can create projects within WSAA to save the results of your exploration. You can then use these projects to task out the work to your application development

team and track their progress. Your application development team can also use these projects to help develop test cases for their changes.

With the power to explore how the components of your application fit together, you can gain a better understanding of how you can maintain or improve your application. The ease with which you can generate a list of components that are affected by a change can reduce your project determination time and resulting maintenance (or development) costs.

Connector information

You can also use WSAA to obtain and build connector information for your mature CICS and IMS programs. You can:

- ▶ Quickly gather input and output data structures of transactions.
For each transaction, WSAA generates:
 - A summary report
 - A COBOL copybook containing the input/output data structures
- ▶ Put this information into a form that you can import into a connector-building tool, such as WebSphere Studio Application Developer.

Help

WSAA also provides you with extensive online help information:

- ▶ Product overview
- ▶ Reference
- ▶ Tutorials
- ▶ Product setup
- ▶ Inventory collection for z/OS and distributed assets
- ▶ WSAA manuals in PDF format
- ▶ Links to related Web sites
- ▶ Glossary and index

Terminology

There are several unique words and phrases that are associated with WebSphere Studio Asset Analyzer. We offer a brief discussion of some of the more common ones:

- ▶ Site
- ▶ Application

- ▶ Concatenation set
- ▶ Asset (also known as artifact)

Each of these terms is described in the following sections.

Site

In the WSAA online help, you can find the word *site* defined as “an enterprise system that users can access through either an HTTP IP address or a functional subsystem IP address.”

This definition is quite formal and, we believe, limited in scope. A site can, perhaps, be better explained as:

The focal point or name by which an organization wants to be known to accommodate that organization’s production application source code.

Example:

Elixir Foodstuffs (a fictional company) has 18 data centers operating worldwide. Each data center runs a z/OS system that supports one or more unique organizational units and has one or more logical partitions (LPARs).

If Elixir Foodstuffs installed only one copy of WSAA for use by all of the data centers, there would be only one site, and it could be defined as “ELIXIR.” However, if the company installed one copy of WSAA in each data center, the site would be defined based on either the locale or the organizational unit (for example, USA43 or SEASIA02).

During inventory collection, WSAA associates the site with each of the components that are scanned into the database. It then organizes the components based on a hierarchy, which is built as a tree structure below the declared site.

Application

WSAA online help defines an application as “a user-defined grouping of components. You can assign CICS transactions, IMS transactions, and members as part of an application.”

An application is another level that WSAA builds as part of its hierarchy. An application name exists below the site name in the logical hierarchy. The application entry represents any grouping of source, JCL, CICS, and other

components that you choose. It is possible to build application names in WSAA that include cross-application groupings as needed by your site's requirements.

For our purposes, an application is a group of components that represent a corporate entity. Your site might have production data sets that contain a variety of different applications' code, or you might have specific data sets for each application. In either case, most applications invariably use some kind of mnemonic device to indicate their function, for example: Payroll systems might use PYR or PAY as the first three characters of each member name. Bookkeeping systems might use BKP as the first three characters of their member names. Each site undoubtedly has its own standards.

WSAA gives you the ability to define components and associate them with these corporate entities either when you load the database or afterward. WSAA even allows you to create hierarchies of applications, for example, you might have a Client Billing subsystem within the Major Fixed Accounts system. When you create an application, WSAA has a field that lets you define a "parent" application. As long as you define the parent applications before the child applications, you can use the hierarchy when you explore the database for relationships (or to correct errors following inventory collection).

Concatenation set

Again, we refer to WSAA's online help, which defines a concatenation set as "an ordered list of libraries to be searched to resolve references to included source (for example, JCL PROCs, assembler macros, or COBOL copybooks)."

During inventory collection, WSAA attempts to resolve the location of each of the included components that are found in main programs. An *analysis concatenation set* — or simply, concatenation set — is a specific group or pool, of libraries, which has a unique name, that can be used to shorten the analysis phase when the source code is scanned.

A concatenation set is the equivalent of the data sets that are used in a SYSLIB DD in a batch compile or specified in the ORDER parameter of a JCLLIB statement. In a similar manner, WSAA uses a top-down search through the libraries that are defined in the concatenation set to determine the appropriate member to use.

Asset

WSAA's online help defines an asset as "a programming asset, such as a file, an object, documentation, or code."

Some earlier WSAA documentation refers to artifacts; however, the current documentation refers to assets. We prefer to use the word *components*.

Irrespective of how you refer to them, these are the essential elements that comprise your application portfolio. They include, but are not limited to:

- ▶ Programs
- ▶ Run units
- ▶ Batch jobs
- ▶ CICS transactions
- ▶ Data sets
- ▶ Data stores
- ▶ DD names

WSAA even considers applications, concatenation sets, and projects to be assets because they also make up the information that relates to your application portfolio.

Using WebSphere Studio Asset Analyzer with CICS IA Explorer

If you have WebSphere Studio Asset Analyzer installed, you can access it in the CICS IA Explorer using the Asset Details option.

To use WSAA to analyze query and search results:

1. Click **Window** → **Preferences**.
2. In the Preferences page on the left navigation, click **WSAA**.
3. Enter your WSAA server name and port number.
4. Click **Apply**.
5. Click **OK**.
6. Right-click any resource, program, or transaction. If WSAA is relevant for that resource, program, or transaction, the Asset Details option is displayed.
7. Click **Asset Details**. WSAA details about the chosen resource are displayed in your external Web browser.

Related publications

The publications that are listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 277. Note that some of the documents referenced here might be available in softcopy only.

- ▶ *Threadsafe Considerations for CICS*, SG24-6351-02
- ▶ *Implementing CICS Web Services*, SG24-7206-02

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,

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

A

- abend codes
 - AICA 18
- Affinities 105, 152
- Affinities utility 4
- Affinity lifetimes 206
 - activity 206
 - facility 206
 - logon 206
 - permanent 206
 - process 206
 - pseudoconversation 206
 - signon 206
 - system 206
- Affinity relations 205
 - BAPPL 205
 - global 205
 - LINK3270 205
 - LUnicode 206
 - user ID 206
- APAR PK56097 197
- Application exploration 271

C

- CEDA INSTALL 19
- CICS API
 - commands 74
- CICS FEPI
 - commands 75
- CICS global user exit 259
- CICS IA 15
 - address space 263
 - Architecture and components 1
 - CIUALOAD 26
 - CIUANEW 26
 - CIUJ13CR 18
 - CIUJCLCA 15
 - data sets. 13
 - Explorer overview 122
 - monitor 74
 - trace 263
 - transactions CINT, CINB, and CINC 17
 - variables 14

- VSAM files 14
- CICS IA Collector 12
 - CINB 77
 - CINT 77
 - global user exit program 77
- CICS IA customization function 12
- CICS IA debugging
 - CICS IA Trace Facility 263
 - CINT Extrapartition 260
 - CINT log 259
 - CIUALOAD 258
 - CIUDBCR 258
 - CIUDBND 258
 - Collector component 258
 - Dependency database 260
 - exit programs 260
 - installation and customization 258
 - Plan Table 265
 - taking a dump of CICS IA 264
- CICS IA Explorer 121
 - Install 46
 - supplied queries 128
- CICS IA explorer
 - Adding filters to the query 137
 - Analyzing query and search results 142
 - Comparing query and search results 144
 - Creating a query folder 141
 - Modifying an existing query 139
 - Programs and Transactions windows 125
 - Regions Window 127
 - Saving your queries in folders 141
 - Toolbar searches and queries 123
- CICS IA explorer - WSAA integration 148
- CICS IA Explorer - WSAA launch 149
- CICS IA Explorer and WSAA 146
- CICS IA Explorer installation 48
- CICS IA Jobs
 - CIUJCLCA 17
- CICS IA LOAD libraries
 - SCIULOAD 18
 - SCIULODE 18
- CICS IA pitfalls
 - cloned systems 246
 - migration 232

CICS IA query 12
 CICS IA requirements 12
 CICS IA Scanner
 detailed report 61
 summary report 53
 CICS IA Trace Facility 263
 CICS Interdependency Analyzer
 Collector component 8
 CINT transaction 77
 components of CICS IA? 6
 enhancements in CICS IA V1R3 5
 installation and customization 11
 Query component 9
 Reporter component 8
 Scanner component 7
 detailed report 61
 summary report 53
 CICS INVOKE WEBSERVICE 196
 CICS SIT
 START=INITIAL 19
 CICS TCB 184
 CICS Transaction Summary view 148
 CICS variables 15
 CINC screen 92
 CIU_AFF_CMD_DATA 152
 CIU_AFF_GRP_DATA 152
 CIU_CICS_DATA 89, 156
 CIU_CICS_DATA table
 AFFINITY 157
 APPLID 156
 CHANNEL 158
 COMMAREA 157
 FIRST_RUN 158
 FUNCTION 157
 HOMESYSID 156
 LAST_RUN 158
 OBJECT 157
 OBJLENGTH 157
 OFFSET 157
 PROGLEN 157
 PROGRAM 156
 RMTNAME 157
 RMTSYSID 157
 TCBMODE 157
 TERMTRAN 157
 TRANSID 156
 TYPE 157
 USECOUNT 158
 CIU_CICS_DATA table. 196
 CIU_DB2_DATA 89, 167
 CIU_EXIT_INFO 175
 CIU_FILE_DETAIL 176
 CIU_IMS_DATA 89, 172
 CIU_MQ_DATA 89, 173
 CIU_PROGRAM_DETAIL 188
 CIU_RESOURCE table 33
 CIU_SCAN_DETAIL table 152
 CIU_SCAN_SUMMARY 180
 CIU_WEBSERV_DETAIL table. 196
 CIU7017I No SQL available for this query 258
 CIUADESC 29
 CIUALOAD 27
 CIUANEW 26
 CIUCNFG1 13
 CIUDBND 23
 CIUGRANT 23
 CIUIVPLD job 30
 CIUJCLCC 15
 CIUJCLLD
 creates a detailed report. 54
 CIUJCLLS
 creates a summary report. 54
 CIUJCLTD
 create a detailed report with DB2 output. 54
 CIUJCLTS
 creates a summary report with DB2 output. 54
 CIUUPDB 33–34, 39, 41, 43
 loads CICS,WMQ,IMS and DB2 resource information. 89
 CIUUPDB1
 loads CICS resource information 89
 CIUUPDB2
 loads DB2 resource information 89
 CIUUPDB3
 loads WMQ resource information 89
 CIUUPDB4
 loads IMS resource information 89
 CIUV transaction 30
 ClearCase 267, 270
 Cloned systems 246
 COBOL/BMS 9
 COIO 108
 CONL 108
 Connector information 272
 CPSM group definitions
 Affinities Reporter 215
 Affinity lifetimes 206
 Affinity relations 205

- customizing and running the Builder 223
- intertransaction affinity 204
- overview of transaction affinities 204
- transaction-system affinity 205
- uploading the data into CICSplex SM 226

CSECT Scanner 67

D

- DB2 calls 5
- DB2 location 49
- DB2 output 63
- DB2 variables 14
- DB2CONN 18
- DB2CONN=YES 18
- DB2ENTRY 18
- Dependencies 156
- detailed report 62
- DETAILMODS 54
- DFH\$PIEX, 197
- DFHEIDTH 188
- DFHEISUP 188
- DFHRPL 18
- DPL 197
- DSWFORVV 99, 102
- DSWSUBLX 94
- Dynamic COBOL calls to other programs 5
- dynamic routing 4

E

- EXEC CICS
 - commands 76
- EXEC CICS ADDRESS CWA 189
- EXEC CICS commands 6
- EXEC CICS INVOKE WEBSERVICE 196
- EXEC CICS WRITE FILE(4
- EXEC CICS, 188
- EXEC DLI 5
- EXEC SQL 188
- Exits 82

F

- file-owning region (FOR) 12
- Files 82
- Freeman 108
- Function shipping t 12

G

- General variables 15
- Getmain 108

H

- Hardware Requirements
 - 9
- Help 272
 - glossary and index 272
 - inventory collection for MVS and distributed as-sets 272
 - links to related Web sites 272
 - product overview 272
 - product setup 272
 - reference 272
 - tutorials 272
 - WSAA manuals in PDF format 272

I

- IMS section of job CIUUPDB 43
- Inventory collection 270
 - distributed 270
 - mainframe 270
 - scanning results 271

J

- JCL
 - CIUADESC 29
 - CIUALOAD job 28
 - CIUANEW job sample 27
 - CIUDBCR sample job 20
 - CIUIVPLD job sample 31
 - SCIUSQL member CIUGRANT 26

L

- Load Module Scanner 152

M

- main Explorer menu 124
- mainframe 270
- Migration assistance 183
- Migration variables 15
- MQ Section of job CIUUPDB 41
- MVS resources 267
 - CICS region information 267
 - IMS 267

JCL 267
source code 267

O

Open Transaction Environment (OTE) 184
OPENAPI-enabled 192
OS/390 Version 2.10 12
overview of WebSphere Studio Asset Analyzer
 graphic overview 269

P

PIPELINE 195
PLT 19
PROGAAAF 125
PROGAAFA 125
PROGAFAA 125
Programs 82
PTF UK33060 197

Q

Quasi-reentrant (QR)
 programs 185
quasi-reentrant (QR) 184
Queries window
 130

R

RACF group CIUGROUP. 25
Redbooks Web site 277
 Contact us xiii
Resource types
 ABEND 158
 BRFACIL 158
 CHANNEL 158
 CONFRMTN 158
 CONTAINR 159
 COPY 159
 CORBASRV 159
 COUNTER 159
 DB2ENTRY 159
 DB2TRAN 159
 DISCONNT 159
 DJAR 159
 DOCTEMP 160
 ENQNAME 160
 ERROR 160
 EXIT 160

FEPI 160
FEPINODE 160
FEPIPOOL 160
FEPISET 160
FEPITGT 161
FILE 161
HANDLE 161
IPCONN 161
JOURNAL 161
JVMPROF 161
LIBRARY 161
MAP 162
MAPSET 162
PASS 162
PIPELINE 162
POOL 162
PROCESS 162
PROGRAM 162
RESET 162
SIGNAL 162
STORAGE 162
STORSHR 163
TCPIPSRV 163
TD 163
TERMINAL 163
TEXT 163
TRANSID 163
TS 163
TSAU 163
TSMODEL 164
TSPOOL 164
TSSHR 164
UOW 164
URIMAP 164
WEB 164
WEBSRV 164

S

Sample Queries 155
scanners 268
SCIUEXEC library 13
SDLC 268
server address 49
Software Requirements
 9
SQLCODE -905 258
summary report 54

T

TCP/IP port number 49
TDQueues 82
terminology 272
 Application 273
 asset 274
 batch jobs 275
 CICS transactions 275
 data sets 275
 data stores 275
 DD names 275
 programs 275
 run units 275
 concatenation set 274
 site 273
Threadafety analysis 184
threadafety analysis 193
Transactions 82
Transaction-system affinity 205
TRUEs 184
TRUEs and GLUEs 201
TSQueues 82

U

Uniform Resource Identifier (URI) 195
URIMAP 195

V

V_CIU_AFFINITY 152
V_CIU_AFFINITY view 152
 AFFINITY 153
 AFFTYPE 153
 AFFWORSENEED 153
 APPLID 152
 BTS 155
 BUILD 155
 COMMAND 155
 GROUPTYPE 153
 LIFETIME 154
 LIFEWORSENEED 154
 LINK3270 155
 OFFSET 155
 PROGCOUNT 154
 PROGRAM 155
 RECOVERY 154
 RESLENGTH 154
 RESOURCE 154
 RESTYPE 155

 TERMINAL 155
 TRANCOUNT 154
 TRANGROUP 152
 TRANSID 155
 TYPE 154
V_CIU_CICS_INDS 165
V_CIU_SCAN_TRDSAFE 188
VSAM files 8
 CIUAFF1 18
 CIUAFF3 18
 CIUCNTL 18
 CIUINT1 18
 CIUINT6 18
VSAM record-level sharing (RLS) 12

W

Web service description language (WSDL) 195
Web services provider detection 197
Web services requester detection 196
WEBSERVICE, 195
Webservices 82
WebSphere Studio Asset Analyzer
 See WSAA
WebSphere Studio Asset Analyzer (WSAA) 146
WSAA 267–275
WSAA assists 268
 analyzing the impact of a proposed change 268
 application developers 268
 deployment phase 268
 development phase 268
 extracting business logic from existing code 268
 gathering connector information for MVS programs 268
 project managers 268
 quality assurance testers 268
 requirements phase 268
 scope and develop project plans 268
 test phase 268
 understanding components and their relationships 268
WSAA core functions 269
 application exploration 270
 Connector building 270
 help 270
 impact analysis 270
 inventory collection 270
WSAA Explorer 147

X

XEIN 197
XEIOUT 197

Z

z/OS 8, 18



IBM CICS Interdependency Analyzer

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



IBM CICS Interdependency Analyzer

CICS Interdependency Analyzer Explorer

The IBM CICS Interdependency Analyzer (CICS IA) is a runtime tool for use with CICS Transaction Server for z/OS. It has two main purposes:

- ▶ To identify the sets of resources that are used by individual CICS transactions and their relationships to other resources.
- ▶ To collect and analyze data about transaction affinities. Transaction affinities require particular groups of transactions to be run either in the same CICS region or in a particular region.

CICS Interdependency Analyzer Threadsafe report

In this IBM Redbooks publication, we first provide a detail overview of what CICS IA is and what business issues it addresses before we review the installation and customization of the product.

We discuss the scanner and collector components in detail, and we then focus on the query interface and the new CICS IA Explorer.

CICS Interdependency Analyzer Analysis of affinity data

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks