

CICS Interdependency Analyzer for z/OS



User's Guide and Reference

Version 3 Release 2

CICS Interdependency Analyzer for z/OS



User's Guide and Reference

Version 3 Release 2

Note!

Before using this information and the product it supports, read the general information under "Notices" on page 383.

This edition applies to Version 3 Release 2 of the IBM CICS Interdependency Analyzer for z/OS, program number 5655-U86, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

© Copyright IBM Corporation 1994, 2011.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	vii	Chapter 2. Getting started with CICS IA 29
Tables	ix	Running the configuration utility 29
Preface	xiii	Customizing your CICS regions to use the CICS IA Collector 30
Who this information is for	xiii	Creating VSAM files 30
How to use this information	xiv	Defining resources to CICS 30
What's new in this version	xv	Tailoring your CICS startup job. 31
Summary of changes for earlier versions	xvii	Collecting DB2 resources in your CICS region . . 31
Changes for Version 3 Release 1	xvii	Restarting your CICS region. 32
Changes for Version 2 Release 2	xviii	Customizing DB2 Environment. 32
Changes for Version 2 Release 1	xx	Running the Installation Verification Program . . . 33
Changes for Version 1 Release 3	xxi	Collecting data with CICS IA 33
Changes for Version 1 Release 2	xxii	Configuring the Collector. 33
Chapter 1. Overview of the CICS Interdependency Analyzer	1	Running the Collector 34
CICS IA requirements	2	Loading CICS IA data 34
CICS IA interdependency functions.	3	Viewing Data with CICS IA Explorer 35
CICS IA dependency-related components overview	4	Chapter 3. Preparing to use CICS IA . . . 37
Dependency-related commands	6	Data sets used during this configuration. 37
CICS IA affinity related functions	6	Running the installation customization program . . 44
What are transaction affinities?	7	Starting CICS IA customization. 45
CICS IA affinity-related components: overview	10	Creating VSAM files 48
Affinity-related commands	12	Creating Log Streams and GDGs 50
CICS IA Command Flow functions	14	Editing and running the CIUJCDLS job 50
CICS IA Command Flow components overview	14	Editing and running the CIUJCLCG job 50
The Command Flow Feature	14	Building the CICS IA database 50
The Collector component	15	Defining the database 51
What can be monitored	17	DB2 Stored Procedures setup 53
What is not monitored.	17	Grouping transactions and programs into applications 54
Controlling the Collector	18	Loading static DB2 tables. 55
How dependency data is collected.	19	Creating your own program exclude, transaction exclude, and resource prefix lists 55
How affinity data is collected	20	Creating a program exclude list 55
Saving data	20	Creating a transaction exclude list 56
The dependency data and affinity data VSAM files	21	Creating a resource prefix list 57
The control record VSAM file	22	Creating your own TRUE include list. 58
The Dependency database objects	23	Creating a TRUE include list 59
The Affinity database objects	23	Migrating from CICS IA Version 3.1 60
The CICS IA plug-in for CICS Explorer	24	Removing old definitions. 60
CICS IA reports	24	Migrating the DB2 tables 60
The Dependency Reporter	24	Migrating the VSAM control record file 61
The Affinities Reporter	25	Migrating the VSAM collection datasets 61
The Threadsafe Reporter	25	Migrating application definitions 61
The Scanner component	26	Defining resources to CICS 61
The Load Module Scanner	26	Installing resource definitions 62
The CSECT Scanner	27	Collecting DB2 resources in your CICS region . . 62
The Builder component	27	CICS IA Natural support 63
		Installing Natural support 63
		Customizing the CICS IA Natural Interface. . . 64
		Security considerations 65
		Tailoring your CICS startup job. 65
		Starting and stopping CICS IA from the PLT . . 65
		Restarting your CICS regions 66

CICS IA supplied modules required in the MVS linklist	66	Output from the Dependency Reporter	128
Chapter 4. Running the installation verification program	67	Running the Affinities Reporter	133
About the IVP	67	Requesting a report from the Affinities Reporter	134
Running the IVP.	67	Output from the Affinities Reporter	134
Loading IVP sample data.	68	Running the affinity report	138
Viewing the IVP sample data	68	Compressing affinity data	140
Chapter 5. Running the Collector	71	Chapter 9. Running the Program Threadsafe report	141
Running the Collector for the first time	71	Analyzing the program dynamic analysis threadsafe report	141
Displaying the Collector Main Administration Menu panel	72	Chapter 10. Running the Load Module Scanner.	145
The CINC transaction User Administration	73	Creating a summary report.	145
Controlling the collection of dependency and affinity data	80	Creating a summary report with DB2 output	147
Starting the collection of data	81	Creating a detailed report	148
Changing the data collection options dynamically	84	Contents of a detailed report	149
Pausing the collection of data	85	Creating a detailed report with DB2 output	151
Resuming the collection of data	86	Chapter 11. Running the CSECT Scanner.	153
Stopping the collection of data	87	The CIUJCLCS job.	154
Displaying Collector statistics for a specified region	89	Contents of the printed report.	154
Changing the Collector options.	90	Chapter 12. Running the Builder	159
Specifying region-specific options: region configuration	90	Editing the CIUAFFBL job	159
Specifying Resource options: region configuration	92	Syntax for input to the Builder	160
Specifying region-specific options: general	93	HEADER statements	162
Specifying region-specific options: API and SPI commands to be monitored	98	Output from the Builder.	163
Specifying which dependency-related DB2, IMS, MQ commands and TRUES are to be monitored	100	Combined affinity-transaction-group definitions	163
Specifying which affinity-related CICS commands are to be monitored	102	Data sets processed report	165
Specifying region-specific options: timers	103	Empty transaction groups report	166
Managing Command Flow data collection.	105	Group merge report	166
Specifying Natural options	112	Error report	167
Changing global options.	114	Chapter 13. Running the sample DB2 query	169
Removing duplicate resources from the Collector files	115	The CIUJSAMP job	169
Operating CICS IA Collection from the CICS IA Explorer plug-in	118	Tailoring the job for your environment	169
Collector errors.	119	Running SPUFI.	169
Chapter 6. Updating the Dependency and Affinity database objects	121	Chapter 14. Solving problems	171
Updating the Dependency database objects	121	Overview of CICS IA problem determination.	171
Database update procedure.	122	Dealing with errors	171
Updating the Affinity database objects	122	Preliminary checks	171
Updating the Command Flow database objects	122	Classifying the problem	172
Chapter 7. Creating and updating a CICS IA UDB database	123	Supplying a CICS IA trace	172
Preparing CSV files	123	Taking a dump of CICS IA	172
Chapter 8. Running the Reporter	125	CICS IA plug-in for CICS Explorer level	174
Running the Dependency Reporter	125	Obtaining an error log	175
Modifying a Dependency Reporter Job	125	Obtaining configuration details	175
		Viewing Eclipse plug-ins	175
		Contacting IBM Support.	176
		When to contact the Support Center.	176
		Working with the Support Center	176
		Information data sheet	177

Appendix A. Details of dependencies and affinities collected 179

Commands monitored for potential dependencies	179
CICS commands detected	179
CICS FEPI commands detected	190
Non-CICS API commands detected	191
Commands monitored for potential affinities	192
CICS API commands	192
CICS SPI commands	193
Details of what is detected	193
ENQ/DEQ	193
TS commands	194
LOAD HOLD/RELEASE	194
RETRIEVE WAIT/START	195
ADDRESS CWA	195
GETMAIN SHARED/FREEMAIN	195
LOAD/FREEMAIN	196
CANCEL/DELAY/POST/START	196
SPI commands	197
WAIT commands	197

Appendix B. Correlating Load Module Scanner and Dependency Reporter output to source 199

Dependency Reporter output	199
Load Module Scanner output	199
Examples	199

Appendix C. The structure of the CICS IA database 203

The structure of the Dependency database objects	203
Dependency base tables	203
Dependency facilitating tables	208
Threadsafe table	209
Dependency views	210
The structure of the Affinity objects	211
Affinity base tables	211
Affinity facilitating table	214
Affinity views	214
The structure of the Load Module Scanner objects	217
Load Module Scanner base tables	217
The structure of the CSECT Scanner objects	219
CSECT Scanner base tables	219
CSECT Scanner object views	220
The structure of the CICS regions objects	221
CICS regions base table	221
The structure of the resource objects	222
File resource table	222
Program resource table	224
Transaction resource table	225
Transient Data queue resource table	228
Temporary Storage queue resource table	230
Web service resource table	231
GLUE and TRUE exit resource table	232
Event table	233
The structure of the CICS IA plug-in for CICS Explorer resource objects	234
The structure of the Version objects	235
The structure of the Command Flow table objects	235

Type and Function mapping for monitored commands	238
--	-----

Appendix D. Messages and codes . . . 255

Contacting IBM Support	255
Messages that CICS IA can issue	255
Collector table manager diagnostics	314
Function code values	314
Table identifier values	314
Reason code values	315
Collector CINB request queue manager diagnostics	316
Function code values	316
Reason code values	316
Date formatter diagnostics	316
Reason code values	316

Appendix E. Worksheet for the installation customization program . . 317

Appendix F. CICS IA space considerations 327

Required data	327
Data space allocation	328
Calculating the space required for interdependency collection	328
Calculating the space required for affinity collection	331
VSAM data set allocation	331
Control file: CIUCNTL	331
Dependency files: CIUINT1, 2, 3, 4, 5, 6, 7	332
Affinity files: CIUAFF1,2,3	332
DB2 space allocation	333
CICS tables and index: CIUCICS1 and CIUCICSX	334
DB2 tables and index: CIUDB2	334
MQ tables and index: CIUMQ1	335
IMS tables and index: CIUIMS	335
Natural tables and an index: CIUNAT	336
Exit resource tables and index: CIUREXIT	337
File resource tables and index: CIURFILE	337
Program resource tables and index: CIURPROG	338
Transaction resource tables and index: CIURTRAN	338
Transient data queue resource tables and index: CIURTDQ	339
Temporary storage queue resource tables and index: CIURTSQ	339
Web services resource tables and index: CIURWEB	340
Affinity tables and indexes	340
Load Module Scanner tables	341
CSECT Module Scanner tables	342
CICS region tables	342
CICS IA plug-in for CICS Explorer Resource table: CIURESTB/X	343

Appendix G. CICS IA security 345

CICS IA transaction security	345
DB2 security	345

Appendix H. How IA affects performance	347	CIUSPAPP Stored Procedure	359
Number of interdependency records per workload	348	CIUSPAFF Stored Procedure	362
Number of affinity records per workload	349	The CICS IA Command Flow user exit	365
Results for interdependency data collection	349	Appendix J. Bibliography	371
Results for affinity data collection	350	CICS Transaction Server	371
How collection data is saved	351	DB2	371
CICS IA performance figures running the CICS		Appendix K. Accessibility	373
DB2 application	352	Enabling hover help for screen readers	373
CICS IA performance figures running the CICS		Index	375
VSAM application	353	Notices	383
Response times	353	Trademarks	384
CICS IA exits	354	Sending your comments to IBM	385
Controlling Collector performance	355		
Appendix I. CICS IA External			
Interfaces	359		
DB2 Stored Procedures	359		

Figures

1.	The Collector structure of CICS IA components	4	34.	Collector Natural Options screen, CIU29N	113
2.	The reporting structure of CICS IA dependency-related components	5	35.	Collector Global Options Menu panel, CIU300	114
3.	The reporting structure of the CICS IA affinity-related components	11	36.	Example clean up utility report	118
4.	Command Flow option structure	15	37.	Configuring Explorer plug-in variables	119
5.	Collector components	16	38.	Example CICS report from the Dependency Reporter—header page	129
6.	Panel CIUCNF04.	46	39.	Example CICS report from the Dependency Reporter—main body	130
7.	Panel CIUCNF05.	47	40.	Example DB2 report from the Dependency Reporter	131
8.	Panel CIUCNF06.	47	41.	Example MQ report from the Dependency Reporter	132
9.	Panel CIUCNF07.	48	42.	Example IMS report from the Dependency Reporter	133
10.	IVP application	54	43.	A sample report output by the Affinities Reporter	135
11.	Example program exclude list	56	44.	Sample basic affinity-transaction-group definitions	138
12.	Example transaction exclude list	57	45.	Example program dynamic analysis threadsafe report, header page	142
13.	Example resource name prefix list	58	46.	Example programs dynamic analysis threadsafe report, main body	142
14.	Example TRUE include list	59	47.	Example of a summary report produced by the Load Module Scanner	147
15.	Collector Main Administration Menu panel, CIU000	72	48.	Builder input syntax	162
16.	User Administration Menu panel, CIU400	74	49.	Sample definitions for combined affinity transaction groups	163
17.	Add User Menu panel, CIU410	75	50.	Sample data sets processed report.	166
18.	Copy User Menu panel, CIU420.	77	51.	Example empty Tranguroups report	166
19.	User Details Menu panel, CIU440	79	52.	Sample group merge report.	167
20.	Collector Operations Menu screen, CIU100	81	53.	Sample error report	168
21.	Collector Statistics Menu panel, CIU150	89	54.	Collector Statistics Menu panel, CIU150, showing the CICS IA data space name	173
22.	Collector Region Configuration Menu panel, CIU200	91	55.	Example output from an MVS DISPLAY ACTIVE command, showing a data space name of 00000INT	174
23.	Collector Resource Options panel, CIU290	92	56.	Example of finding an EXEC CICS command from the argument zero	200
24.	Collector General Options panel, CIU260	94	57.	Calculation for PRIQTY for tablespaces and indexes	333
25.	Collector CICS Resources Options panel, CIU240	99	58.	The sample user exit program	369
26.	Collector CICS Resources Options panel, CIU245	100			
27.	Collector DB2/MQ/IMS/RMI True Resource Options panel, CIU250	101			
28.	Collector Affinities Options panel, CIU270	103			
29.	Collector Time and Dates Options screen, CIU280	104			
30.	Command Flow Options panel, CIUA01	106			
31.	Command Flow ApplID list panel, CIUA02	109			
32.	CICS IA Command Flow Statistics panel, CIUA03	110			
33.	List of available CICS regions , CIUA04	112			

Tables

1. Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner	13	40. The CIU_AFF_INDEX_DATA table	214
2. Jobs and files supplied in hlq.SCIUSMP1	37	41. View V_CIU_AFFINITY	214
3. Jobs and files supplied in hlq.SCIUSMP2	39	42. The CIU_SCAN_SUMMARY table	217
4. Files supplied in hlq.SCIUSQL	40	43. The CIU_SCAN_DETAIL table	217
5. CICS IA VSAM files and associated jobs	48	44. The V_CIU_SCAN_TRDSAFE table	218
6. Editing the CIUDELGR job to remove old CICS IA definitions from CICS TS	60	45. The CIU_PROGRAM_INFO table	219
7. The groups of CICS resource definitions	62	46. The CIU_CSECT_INFO table	219
8. The control keys on the User Administration Menu panel	74	47. The CIU_TRANSLATORS table	220
9. The control keys on the Add User Menu panel	76	48. View V_CIU_CICS_LINKED	220
10. The control keys on the Copy User Menu panel	78	49. View V_CIU_CSECT_TRANS	221
11. The control keys on the User Details Menu panel	80	50. The CIU_REGION_INFO table	221
12. Methods for starting data collection by the Collector	81	51. The CIU_FILE_DETAIL table	222
13. Methods for changing data collection options	84	52. The CIU_PROGRAM_DETAIL table	224
14. Methods for pausing data collection by the Collector	85	53. The CIU_TRANSID_DETAIL table	226
15. Methods for resuming data collection by the Collector	86	54. The CIU_TDQUEUE_DETAIL table	228
16. Methods for stopping data collection by the Collector	87	55. The CIU_TSQUEUE_DETAIL table	230
17. The User Command Flow Utility control keys on the Command Flow Options panel	108	56. The CIU_WEBSERV_DETAIL table	231
18. The User Command Flow Utility control keys on the Command Flow ApplID list panel	109	57. The CIU_EXIT_INFO table	232
19. The User Command Flow Utility control keys on the User Command Flow Statistics panel	111	58. The CIU_TRUEEXIT_INFO table	233
20. The User Command Flow Utility control keys on the CIUA04 panel	112	59. The V_CIU_TRUEEXIT_INFO table	233
21. Resulting affinity relations	164	60. The CIU_EVENT_DETAIL table	233
22. Resulting affinity lifetimes (LUNAME relation)	164	61. The CIU_RESOURCE table	235
23. Resulting affinity lifetimes (BAPPL relation)	165	62. CIU_VERSION table	235
24. Resulting affinity lifetimes (USERID relation)	165	63. The CIU_CMDFLOW_DATA table	236
25. Resulting affinity lifetimes (LINK3270 relation)	165	64. The CIU_CMDFLOW_INDEX table	237
26. Resulting affinity lifetimes (GLOBAL relation)	165	65. Type and Function mapping for monitored commands using the API ATOMServices CICS resource option flag	239
27. Data sheet of problem determination information for IBM Support Center	177	66. Type and Function mapping for monitored commands using the SPI ATOMServices CICS resource option flag	239
28. The CIU_CICS_DATA table	203	67. Type and Function mapping for monitored commands using the SPI BRFacility CICS resource option flag	239
29. The CIU_DB2_DATA table	205	68. Type and Function mapping for monitored commands using the SPI Bundles CICS resource option flag	239
30. The CIU_IMS_DATA table	206	69. Type and Function mapping for monitored commands using the SPI Corbaserver CICS resource option flag	239
31. The CIU_MQ_DATA table	206	70. Type and Function mapping for monitored commands using the API Counters CICS resource option flag	240
32. The CIU_NATURAL_DATA table	207	71. Type and Function mapping for monitored commands using the SPI CSD CICS resource option flag	240
33. The CIU_APPLS_DESC table	209	72. Type and Function mapping for monitored commands using the SPI DB2 CICS resource option flag	242
34. The CIU_APPLS_RESOURCES table	209	73. Type and Function mapping for monitored commands using the SPI DJAR CICS resource option flag	243
35. The CIU_THREADSafe_CMD table	209	74. Type and Function mapping for monitored commands using the API EVENT proc CICS resource option flag	243
36. View V_CIU_CICS_INDS	210		
37. View V_CIU_DB2_RES	211		
38. The CIU_AFF_GRP_DATA table	211		
39. The CIU_AFF_CMD_DATA table	213		

75. Type and Function mapping for monitored commands using the SPI EVENT proc CICS resource option flag	243	95. Type and Function mapping for monitored commands using the SPI TCPIPService CICS resource option flag	248
76. Type and Function mapping for monitored commands using the API Exits CICS resource option flag	243	96. Type and Function mapping for monitored commands using the API TD Queues CICS resource option flag	248
77. Type and Function mapping for monitored commands using the SPI Exits CICS resource option flag	243	97. Type and Function mapping for monitored commands using the SPI Temp Storage CICS resource option flag	249
78. Type and Function mapping for monitored commands using the FEPI API CICS resource option flag	244	98. Type and Function mapping for monitored commands using the API Transactions CICS resource option flag	249
79. Type and Function mapping for monitored commands using the FEPI SPI CICS resource option flag	244	99. Type and Function mapping for monitored commands using the SPI Transactions CICS resource option flag	249
80. Type and Function mapping for monitored commands using the SPI File CICS resource option flag	244	100. Type and Function mapping for monitored commands using the SPI Transient Data CICS resource option flag	249
81. Type and Function mapping for monitored commands using the API Files CICS resource option flag	245	101. Type and Function mapping for monitored commands using the API TS Queues CICS resource option flag	249
82. Type and Function mapping for monitored commands using the SPI IPCONN CICS resource option flag	245	102. Type and Function mapping for monitored commands using the API Web Services CICS resource option flag	250
83. Type and Function mapping for monitored commands using the API Journals CICS resource option flag	245	103. Type and Function mapping for monitored commands using the SPI Web Services CICS resource option flag	250
84. Type and Function mapping for monitored commands using the SPI Journals CICS resource option flag	245	104. Type and Function mapping for monitored commands using the API WSAddressing CICS resource option flag	251
85. Type and Function mapping for monitored commands using the SPI JVMServer CICS resource option flag	246	105. Type and Function mapping for monitored commands using the API XMLTransform CICS resource option flag	251
86. Type and Function mapping for monitored commands using the SPI Library CICS resource option flag	246	106. Type and Function mapping for monitored commands using the SPI XMLTransform CICS resource option flag	251
87. Type and Function mapping for monitored commands using the SPI MQCONN CICS resource option flag	246	107. The possible combinations of TYPE and FUNCTION values in DB2 queries	251
88. Type and Function mapping for monitored commands using the API Others CICS resource option flag	246	108. The possible combinations of TYPE and FUNCTION values in IMS queries	254
89. Type and Function mapping for monitored commands using the API Presentation CICS resource option flag	247	109. The possible combinations of TYPE and FUNCTION values in MQ queries	254
90. Type and Function mapping for monitored commands using the API Presentation or the API DTP CICS resource option flag	247	110. The possible combinations of TYPE and FUNCTION values in Natural queries	254
91. Type and Function mapping for monitored commands using the API Presentation or the API Others CICS resource option flag	247	111. Variables that can be passed to the installation customization program	318
92. Type and Function mapping for monitored commands using the SPI Programs CICS resource option flag	247	112. Names that are not customized by the installation customization program	325
93. Type and Function mapping for monitored commands using the API Programs CICS resource option flag	248	113. Values required for each CICS region	327
94. Type and Function mapping for monitored commands using the API Task Control CICS resource option flag	248	114. Values required for VSAM and DB2 calculations	328
		115. Worksheet for CICS tablespace using DB2 V7.1.	334
		116. Worksheet for CICS tablespace using DB2 V8.1.	334
		117. Worksheet for DB2 tablespace	335
		118. Worksheet for MQ tablespace	335
		119. Worksheet for IMS tablespace	336
		120. Worksheet for Natural tablespace	336
		121. Worksheet for exit resource tablespace	337
		122. Worksheet for file detail resource table	337

123. Worksheet for program detail resource table	338	137. CPU cost of collecting affinity data for some CICS commands, in microseconds 350
124. Worksheet for the TRANSID detail resource table 338	138. CPU consumed per transaction at different transaction rates for a CICS DB2 application 352
125. Worksheet for the TDQUEUE detail resource table 339	139. CPU consumed per transaction at different transaction rates for a CICS VSAM application 353
126. Worksheet for the TSQUEUE detail resource table 339	140. GLUE exits enabled for interdependency data collection option settings. 354
127. Worksheet for the WEBSERV detail resource table 340	141. GLUE exits enabled for affinity data collection option settings. 355
128. Worksheet for Affinity tablespace 341	142. Global user exits used by CICS IA to collect resource dependencies 355
129. Worksheet for Load Module Scanner tablespace. 341	143. Global and task-related user exits used by CICS IA to collect transaction affinities 356
130. Worksheet for CSECT Module Scanner tablespace. 342	144. Global and task-related user exits used by CICS IA to collect Command Flow data. 357
131. Worksheet for CICS region tablespace	343	145. CIUSPAPP parameters 359
132. Worksheet for Resource tablespace using DB2 V7.1. 343	146. calltype values. 360
133. Worksheet for Resource tablespace using DB2 V8.1. 343	147. return-code values 360
134. RACF categories for CICS IA transactions	345	148. CIUSPAFF parameters 362
135. CPU cost of collecting interdependency data for a selection of CICS commands, in microseconds 349	149. Available qarg1 and qarg2 values 363
136. CPU cost of collecting interdependency data for DB2 and WebSphere MQ requests, in microseconds 349	150. qarg1 and qarg2 values in detail 363
		151. rc values 364
		152. sqlcode values 364
		153. User exit return code values 367

Preface

This information describes the IBM® CICS® Interdependency Analyzer. It explains what the program does and how to set up and run its various components.

What this information is about

CICS Interdependency Analyzer (CICS IA) is a run time tool for use with CICS Transaction Server for z/OS®. It has three main purposes:

1. To identify the sets of resources used by individual CICS transactions, and their relationships to other resources.

CICS IA enables you to understand the characteristics of your application set, that is:

- what a CICS region contains,
- which resources a transaction needs to be able to run,
- which programs use which resources,
- which resources are no longer used.

Understanding these characteristics improves your ability to maintain, enhance, modify, or redistribute your applications.

CICS IA captures interdependency information while CICS is running and stores it in VSAM files, from which detailed reports can be produced. The VSAM files are used to load DB2® databases, on which SQL queries can be performed.

2. 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.

This function is useful in a dynamic routing environment, you might need to know of any restrictions that prevent particular transactions being routed to particular application-owning regions (AORs) or that require particular transactions to be routed to particular AORs.

CICS IA loads the affinity data into DB2 databases, on which SQL queries can be performed and from which detailed reports can be produced.

3. To collect and analyze Command Flow data for a given transaction or terminal.

The Command Flow feature records all CICS, DB2, MQ and IMS™ commands issued in a chronological order. This allows you to understand the different paths a given transaction can go. You can run your own Command Flow captures and view it through the IA Explorer plug-in.

Who this information is for

This information is for anyone who needs to understand, install, or use the CICS Interdependency Analyzer.

It will be particularly useful to system architects, system programmers, application programmers, and operators in organizations that need to do one or more of the following:

- Reuse existing applications as e-business
- Split the workload to plan for high availability
- Reduce the cost of application maintenance

- Set up or maintain a dynamic routing environment

What you need to know to understand this information

You need to be familiar with the CICS application programming interface (API), SQL, and the various programming techniques available to CICS application programmers.

How to use this information

This information is intended to be read sequentially.

This information will allow you to understand how to:

1. Set up the Analyzer
2. Run the separate components

Later, when you are familiar with the utility, you need only refer to the section dealing with the particular component that you want to run.

What's new in this version

CICS IA V3.2 delivers a wide range of important new capabilities:

Enhanced CICS IA Command Flow Feature

CICS IA V3.2 introduces a new User Command Flow Feature. This allows individual developers to collect Command Flow data, use batch jobs to load their data and the CICS IA Explorer plug-in to view their data. The Command Flow feature can collect data in a chronological order by transaction(s) or terminal. A new transaction, CINC, is used to operate and administer Command Flow runs. You can also operate and administer it from the CICS IA Explorer plug-in.

Operating the Collector through the CICS IA plug-in for the CICS Explorer®

In CICS IA V3.2, you can control the operation of the CICS IA Dependency and Affinity Collector using the CICS IA plug-in for the CICS Explorer. You can START/STOP/PAUSE/CONTINUE and REFRESH the controller.

Enhanced dynamic call support

CICS IA V3.2 now identifies the program invoking CICS/MQ/IMS/DB2 commands, even if this is a dynamically called program.

Enhanced Affinity collection support

In CICS IA V3.2, you can store collected affinity data in both DB2 zOS and DB2 UDB databases. It provides the ability to extract the Affinity data into CSV files and a sample stored procedure to load the data into DB2 for zOS or a UDB database table.

Enhanced MQ API support

CICS IA V3.2 provides the ability to collect MQ V7.1 API commands supported by CICS TS 4.1 and above.

Enhanced installation and customization

In CICS IA V3.2, you can take advantage of the enhanced ISPF configuration support that simplifies CICS IA installation and customization process.

This includes "shared" configuration support. Previously, all configurable variables were stored in a dataset owned by the *userid*. Now, you can save them in a shared dataset.

CICS TRUE mapping support

In CICS IA V3.2, you can use a DB2 table to map the TRUE name to a description. The description will be shown in the detailed information for that TRUE in the Properties view in the CICS IA Explorer plug-in.

Enhanced Application Resource Information

In CICS IA V3.2 Explorer plug-in, you can now discover resources used by Application.

Internal trace

CICS IA V3.2 now uses the CICS TS user trace feature. This allows up to three levels of tracing and helps with CICS IA problem determination.

Data Life Cycle Management

CICS IA V3.2 introduces the ability to identify a CICS Dependency collection at DB2 load time. This enables the user to load, manage, and compare resource usage by collection id.

Summary of changes for earlier versions

The changes made for Version 2.2 and earlier are listed in this section.

Changes for Version 3 Release 1

CICS IA V3.1 delivers this wide range of important new capabilities:

Command Flow feature

CICS IA V3.1 introduces a new feature to capture all CICS, DB2, IMS, and MQ commands in chronological order. The user can define up to five transactions for which data will be captured. The data is written to a *User Journal*, which can be defined on DASD or in a Coupling Facility. The data is subsequently written to a new DB2 table, CIU_CMDFLOW_DATA. The data captured includes TCB swap information, Task IDs, and Units of Work.

The user can also give the trace an 8-character name. The trace name, and the start and end timestamps are written to the journal and subsequently to a new DB2 table, CIU_CMDFLOW_INDEX.

Enhanced Natural and ADABAS support

CICS IA V3.1 introduces new exits to enable the capture of Natural program calls and ADABAS calls in the Natural environment.

This data is written to a new DB2 table, CIU_NATURAL_DATA.

CICS IA plug-in for the CICS Explorer

In CICS IA V3.1 the CICS IA Explorer is now shipped as the CICS IA plug-in for the CICS Explorer™. It includes enhancements so that the user can query more CICS IA tables, including the CIU_CMDFLOW_DATA and CIU_NATURAL_DATA.

DB2 batch jobs

In CICS IA V3.1, the DB2 batch jobs for the dependency and command flow data use LOAD and UNLOAD utilities for better performance.

Support for CSV files

CICS IA V3.1 provides sample jobs to unload the dependency and command flow data to CSV files rather than to DB2 on z/OS. These files can be sent by FTP to other platforms for use with Universal DB2 or spreadsheets.

Enhanced Configuration

The configuration step in CICS IA V3.1 has been enhanced to include more configurable options and to store multiple configurations.

You can customize CICS IA V3.1 for more than one configuration of CICS TS versions and DB2 versions. See “Starting CICS IA customization” on page 45.

The Collector

In CICS IA V3.1, the Collector exits have been reworked to improve performance.

The CINQ transaction

In CICS IA V3.1, the CINQ transaction has been removed.

Information about how IA effects the performance of your system

A new section describing how using CICS IA effects performance has been added. This section describes the performance overhead associated with collecting interdependency and affinity data. See Appendix H, “How IA affects performance,” on page 347 for more information.

Ability to recognize EGL programs

CICS IA V3.1 can now capture and view EGL segments in the scanned load modules.

Time stamps

In version 3.1, time stamps in CICS IA trace records are reflected in the local time format.

Affinity and Dependency issues

CICS IA V3.1 makes it possible to capture both Dependency data and Affinity data at the same time.

Dynamic updating options

You can change the CICS IA V3.1 monitoring options without restarting.

API and SPI commands

CICS IA V3.1 provides an expanded range of API and SPI commands that are supported by the runtime collector. These commands are written to the DB2 table CIU_CICS_DATA.

Logging the Collector options values

In this version, every issued Collector command is written to the CINT log. For the START and REFRESHOPTIONS commands, the list of the Collector runtime options is also written.

All the changed collection options are written to the CINT log after saving to the control file.

Changes for Version 2 Release 2

CICS IA V2.2 delivers a wide range of important new capabilities, including:

CICS IA plug-in

A new CICS IA plug-in for CICS Explorer, (CICS IA plug-in), that you download to a workstation. Using the Eclipse-based CICS IA plug-in, you can run predefined queries provided with CICS IA and create your own queries to analyze CICS IA data. The CICS IA plug-in replaces the client program that was provided with CICS IA V2.1.

When connected to a system containing the DB2 database with CICS IA data, it can be used to explore and analyze that data. The Eclipse-based interface enables effective data presentation. You can use the queries provided with CICS IA or easily build your own that meet your exact need for information. The CICS IA plug-in presents this information in an hierarchical fashion, showing clearly the resource relationships.

New resource tables

CICS IA V2.2 introduces seven new resource tables. They contain detailed information for seven of the primary resource types:

- Transactions
- Programs
- Files
- TD queues

- TS queues
- Web services
- Exits

These tables are repositories for information returned from an EXEC CICS INQUIRE request for the relevant resource type. For example, the File table contains information about the file type.

More detailed information on programs

CICS IA V2.2 collects more detailed information on programs. It also provides sample queries that can help plan for CICS TS version-to-version migration. It can help you see which task-related user exits (TRUEs) and global user exits (GLUEs) are customized by your CICS organization. When you have determined the CICS user exits that have changed in the new version, you can identify which exit programs need testing.

Sample queries

CICS IA V2.2 provides sample queries that tell you which programs include CICS SPIs or APIs that have been removed or changed from one version to the next.

Threadsafe identification

CICS IA V2.2 delivers a DB2 table that indicates, by CICS version, which APIs and SPIs are considered to be threadsafe (that is, they do not perform a swap to the QR TCB). It has been shown that running applications as threadsafe can save as much as 15% of processor usage. This CICS IA table, along with detailed information on programs and files, can be used to produce a report for a given program that will provide information such as:

- Count of threadsafe calls.
- Count of non-threadsafe calls.
- Details of these calls.
- Listings of programs that contain the four EXEC CICS commands that could cause an unsafe affinity between transactions. These commands might need to be resolved for the program to be threadsafe (ADDRESS CWA, LOAD HOLD, GETMAIN SHARED and EXTRACT EXIT).
- Count of MQ, IMS, and DB2 calls.
- Count of dynamic calls.

CICS IA provides the most comprehensive analysis available of threadsafe attributes, so that it provides you with the information you need to assess and modify applications and remove constraints.

Ability to capture new resources

CICS IA V2.2 provides a new table to capture resources used by a Web service. It captures the program name, URIMAP, container, pipeline, mapping levels, WSDL file name, and WSBIND file name. This enables you to understand which resources are required when deploying your Web service from development into test, and from test into production.

CICS IA V2.2 captures resource information for EXEC CICS commands that are considered to be presentation logic. It also captures when the program is called with a COMMAREA or CHANNEL resource. With this information you can identify a legacy program that contains only business logic and could be a candidate to be re-factored into a Web service.

In each new release, CICS IA expands the scope of its data capture. CICS IA V2.2 captures information on the two new resources introduced in CICS

TS 3.2 (IPCONN and LIBRARY). It also captures for the first time information on any EXEC CICS command that has more than one resource associated with it. For example, where an EXEC CICS LINK PROGRAM has a channel associated with it, CICS IA collects both the program name and the channel name. In this case, the program is referred to as the primary resource and the channel as the secondary resource. CICS IA maintains such information about related resources so that the CICS IA plug-in can present a useful hierarchical view of resource relationships.

Additional WebSphere® MQ attributes

CICS IA V2.2 delivers extensions to existing support, now reporting on two additional attributes of the MQOPEN request. These are:

- MQOO_FAIL_IF QUIESCING
- MQOO_BIND_ON_OPEN

Improved program call chain

CICS IA V2.2 reports on all programs involved in a chain of called programs, even those that do not invoke CICS services. Previously it reported only on programs that contained EXEC CICS commands. This enhancement ensures you have a complete record of assets involved in a CICS transaction.

WebSphere Studio Asset Analyzer

CICS IA V2.2 makes it easy to launch WebSphere Studio Asset Analyzer to extend the scope of analysis. An on-screen button in the CICS IA plug-in launches this companion discovery tool.

Enhanced ISPF configuration EXEC

The ISPF configuration EXEC has been updated to allow you to select further DB2 and data management options. You can:

- Use DFSMS data, storage, and management classes to define the VSAM datasets used by CICS IA.
- Customize the size of the VSAM files allocated. For DB2, customize the database name, the storage group name, the plan name for both CICS and batch, and the buffer pools used for table spaces and indexes.

Natural fourth generation language (4GL)

CICS IA V2.2 identifies the use of CICS resources accessed by application programs using Software AG Natural 4GL software. CICS IA V2.2 introduces initial support to identify CICS resources within their Software AG Natural environment:

- CICS programs called from with Natural programs.
- Resources used by the CICS programs called from with Natural programs.
- Calls made to the Adabas task related user exits.

Changes for Version 2 Release 1

The most significant changes for this release are:

- The CINT transaction, used to control the collection of dependency and affinity data, has been enhanced as follows:
 - Start, stop, pause, or resume the collection of data on multiple regions at the same time, that is, with a single command. See “Controlling the collection of dependency and affinity data” on page 80.

- Set a region-specific attribute to the same value on multiple regions with a single command. See “Changing the Collector options” on page 90.
- Set default values for region-specific attributes. See “Changing the Collector options” on page 90.
- You can now set a timer, to control the dates and times at which dependency or affinity data is collected on a region. See “Specifying region-specific options: timers” on page 103.
- You can specify, by means of a list of name prefixes, a set of programs for which data is not to be collected. See “Creating a program exclude list” on page 55.
- You can specify, by means of a list of name prefixes, a set of CICS transactions for which data is not to be collected: see “Creating a transaction exclude list” on page 56.
- A new database table, CIU_REGION_INFO, is introduced, to store information about the CICS regions on which the Collector runs. CIU_REGION_INFO is described in “The structure of the CICS regions objects” on page 221.
- A new installation customization program helps you to customize the CICS IA sample jobs, clists, and SQL definitions. It creates customized installation jobs in which the names of system entities, such as the high-level qualifier (hlq) of the CICS IA data sets, are set to specified values to suit your local environment. The installation customization program is described in “Running the installation customization program” on page 44.
- Appendix E, “Worksheet for the installation customization program,” on page 317 is a new appendix that contains a worksheet for use with the installation customization program.
- A new program, the CICS IA client, which is designed to run in an Eclipse development environment, provides a graphical front end to CICS IA; See “The CICS IA plug-in for CICS Explorer ” on page 24.
- A new Scanner, the CSECT Scanner, is introduced. It scans load modules for information that can be used to identify the version of each CSECT. The output is stored in DB2 tables and can be used to identify different versions of programs. See “The CSECT Scanner” on page 27, Chapter 11, “Running the CSECT Scanner,” on page 153, and “The structure of the CSECT Scanner objects” on page 219.
- New and changed messages are listed in Appendix D, “Messages and codes,” on page 255.
- Chapter 14, “Solving problems,” on page 171 is a new chapter.

Changes for Version 2 Release 1, PTF PK35107

The most significant changes made for this PTF are:

- New Chapter 13, “Running the sample DB2 query,” on page 169
- New Appendix F, “CICS IA space considerations,” on page 327
- New Appendix G, “CICS IA security,” on page 345
- Addition of references to CICS TS V3.2

Changes for Version 1 Release 3

The most significant changes for this release are:

- CICS IA Version 1.3 can now collect and analyze data about transaction affinities, as well as data about resource dependencies. It loads the affinity data into DB2 databases, on which SQL queries can be performed and from which detailed reports can be produced. For more information see the following new sections:
 - “CICS IA affinity related functions” on page 6
 - “How affinity data is collected” on page 20
 - “The Affinity database objects” on page 23
 - “The Affinities Reporter” on page 25
 - “The Load Module Scanner” on page 26
 - “The Builder component” on page 27
 - “Updating the Affinity database objects” on page 122
 - Displaying affinities data
 - “Running the Affinities Reporter” on page 133
 - Chapter 12, “Running the Builder,” on page 159
 - “Commands monitored for potential affinities” on page 192
 - “The structure of the Affinity objects” on page 211
 - “The structure of the Load Module Scanner objects” on page 217
- “Changes for Version 2 Release 1” on page xx summarizes the new features in CICS IA Version 1.3.
- The configuration information in Chapter 3, “Preparing to use CICS IA,” on page 37 has been updated.
- “Migrating from CICS IA Version 3.1” on page 60 has been rewritten.
- Chapter 4, “Running the installation verification program,” on page 67 is a new chapter.
- Chapter 5, “Running the Collector,” on page 71 has been rewritten to include the collection of affinity data.
- New JCL to update the Dependency database objects is described in “Updating the Dependency database objects” on page 121.
- Chapter 10, “Running the Load Module Scanner,” on page 145 has been rewritten. “Creating a summary report with DB2 output” on page 147, “Creating a detailed report with DB2 output” on page 151, and “The structure of the Load Module Scanner objects” on page 217 are new sections.
- New messages have been added to “Messages that CICS IA can issue” on page 255.
- The collection of resource dependencies caused by EXEC DLI calls to IMS databases is documented in:
 - “Non-CICS API commands detected” on page 191
 - Displaying interdependency data for IMS resources
 - “IMS database: EXEC DLI” on page 191
- The name of the specific DB2 resource causing a DB2 dependency is now supplied in the CIU3_DB2_DATA base table. In CICS IA Version 1.2 it had to be looked up in the DB2 SYSIBM.SYSPACKSTMT or SYSIBM.SYSSTMT table. See:
 - Viewing the DB2 resource referenced by a DB2 command
 - The description of the CIU3_DB2_DATA base table in “Dependency base tables” on page 203

Changes for Version 1 Release 2

The book was largely rewritten to describe the new features in this release. The most significant changes were:

- “Dependency-related commands” on page 6 was updated to include additional CICS API, SPI, and FEPI commands that can now be monitored by CICS IA.

- “Non-CICS API commands detected” on page 191 listed the non-CICS commands that can now be monitored by CICS IA.
- The first section of Chapter 3, “Preparing to use CICS IA,” on page 37 listed the revised set of installation files now supplied with CICS IA.
- “DB2 considerations” on page 51 was a new section.
- “Migrating from CICS IA Version 3.1” on page 60 was a new section.
- Chapter 5, “Running the Collector,” on page 71 was rewritten to describe the new interface to the CICS IA Collector.
- “Updating the Dependency database objects” on page 121 was rewritten to describe how to update the new tables in the Dependency database objects.
- ‘Using the CICS IA Client’ was rewritten to describe CICS IA's revised Query interface.
- “Running the Dependency Reporter” on page 125 was rewritten to describe how to generate the new reports that are now available.
- “CICS SPI commands” on page 186 gave details of the dependencies collected for CICS system programming interface (SPI) commands.
- “CICS FEPI commands detected” on page 190 gave details of the dependencies collected for CICS Front End Programming Interface (FEPI) commands.
- “Non-CICS API commands detected” on page 191 gave details of the dependencies collected for non-CICS API commands.
- Appendix C, “The structure of the CICS IA database,” on page 203 was rewritten to describe the structure of the new tables in the set of Dependency database objects.
- “Messages that CICS IA can issue” on page 255 was updated with new and changed CICS IA messages.
- “Collector table manager diagnostics” on page 314 was updated.

Chapter 1. Overview of the CICS Interdependency Analyzer

This section gives an overview of the CICS Interdependency Analyzer (CICS IA), and describes its components.

CICS IA is a run time tool for use with CICS Transaction Server for z/OS. It has three purposes:

1. To identify the sets of resources used by individual CICS transactions and their relationships to other resources. Then you can understand the characteristics of your application set: you can see what a CICS region contains; what resources a transaction needs in order to run; which programs use which resources; and which resources are no longer used. Thus your ability to maintain, enhance, modify, or redistribute your applications is much improved.

This function of CICS IA is described in “CICS IA interdependency functions” on page 3.

2. To identify possible transaction affinities. Affinities require particular groups of transactions to be run either in the same CICS region, or in a particular region. The ability to identify transaction affinities is useful in a dynamic routing environment: you need to know of any restrictions that prevent particular transactions being routed to particular application-owning regions (AORs); or that require particular transactions to be routed to particular AORs.

This function of CICS IA is described in “CICS IA affinity related functions” on page 6.

3. To identify and analyze resource usage flow within a transaction or transactions. This is done using the Command Flow feature. It allows individual users to capture all CICS/DB2/MQ/IMS commands in chronological order. The data is stored in DB2 tables, and each individual user can populate these tables with their own data. The CICS IA Explorer plug-in provides a new view to list all the Command Flow captures by a userid.

This function of CICS IA is described in “CICS IA Command Flow functions” on page 14.

The rest of this section contains:

- “CICS IA requirements” on page 2
- “CICS IA interdependency functions” on page 3
- “CICS IA affinity related functions” on page 6
- “The Collector component” on page 15
- “The Dependency database objects” on page 23
- “The Affinity database objects” on page 23
- “The CICS IA plug-in for CICS Explorer ” on page 24
- “CICS IA reports” on page 24
- “The Threadsafe Reporter” on page 25
- “The Scanner component” on page 26
- “The Builder component” on page 27
- “The Command Flow Feature” on page 14

CICS IA requirements

Requirements for running CICS IA.

CICS Requirements

CICS IA V3.2 captures data for CICS Transaction Server Version 1.3 for z/OS and later. However, some new features are supported only in later versions of CICS TS for z/OS.

The CICS IA Command Flow feature requires CICS TS V3.1 or later.

If you want to operate the CICS IA Controller from the CICS IA Explorer plug-in, the CICS IA region that you intend to control requires CICS Transaction Server V4.1 or later.

Each CICS region, on which the CICS IA Collector is to run, must have Language Environment[®] installed and active.

DB2 requirements

CICS IA requires DB2 V8.1 for z/OS or above. You will also require the *DB2 Utilities Suite for z/OS* for the version DB2 that you are using.

The CICS IA sample jobs support utilities provided by other vendor with minor changes.

VSAM file support

To control CICS IA Collectors on multiple regions from a single CICS terminal, the VSAM files to which CICS saves dependency data, affinity data, and control information must be shared across all the regions. See “The dependency data and affinity data VSAM files” on page 21 and “The control record VSAM file” on page 22. To share these files, you can use either:

1. VSAM record-level sharing (RLS). If you use VSAM RLS, all the regions must be in the same MVS[™] parallel sysplex. A parallel sysplex is a sysplex that uses a coupling facility, which is required to support VSAM RLS. For information about using VSAM RLS in CICS, see the *CICS Installation Guide*.
2. Function shipping to a file-owning region (FOR). For information about CICS function shipping, see the *CICS Intercommunication Guide*.

Other Requirements

CICS IA V3.2 uses the IBM supplied DFSORT utility in the sample batch jobs used to load DB2 tables.

These jobs can be used with other vendor SORT utilities with minor changes to the sample JCL.

Note: The SORT utility uses SYMBOLIC names. CICS IA does not provide support for vendor utilities that do not meet this requirement.

CICS IA interdependency functions

CICS IA assists in understanding, in a controlled manner, the inter relationships between the shared common resources of applications and services.

Many large organizations have been using CICS since the early 1970s, their systems growing and evolving with the business. During this time, many techniques for implementing applications have been used, as a result of new function, changing corporate standards, technical requirements, and business pressures. Frequently, this growth has not been as structured as it might have been, with the result that many applications and services share common resources, and changes in one area typically affect many others. Unstructured growth can reach such a level that the system can no longer develop in a controlled manner without a full understanding of these inter relationships. CICS IA can help you achieve this understanding.

For example, to change the content or structure of a file, you must know which programs use this file, because they will need to be changed. CICS IA can identify the programs and 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 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, and other programs.

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 commands that operate on resources that are not CICS. For example:

- EXEC SQL calls to DB2 Universal Database™ resources
- MQ calls to WebSphere MQ resources
- EXEC DLI calls and language-dependent native calls to IMS Database resources
- Dynamic COBOL calls to other programs

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

All the CICS and non-CICS commands that can be monitored are listed in Appendix A, "Details of dependencies and affinities collected," on page 179.

The Collector component of CICS IA collects the dependencies that apply to a single CICS region; that is, a single application-owning region (AOR) or a single, combined routing region and AOR. It can be run against production CICS regions and is also useful in a test environment, to monitor possible dependencies introduced by new or changed application suites or packages. From the interactive interface of CICS IA, you can control Collectors running 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.

The rest of this section contains:

- “CICS IA dependency-related components overview”
- “Dependency-related commands” on page 6

CICS IA dependency-related components overview

CICS IA comprises a number of components, which divide into collecting and reporting parts. Study these figures to help you understand how the dependency-related components of CICS IA relate to each other.

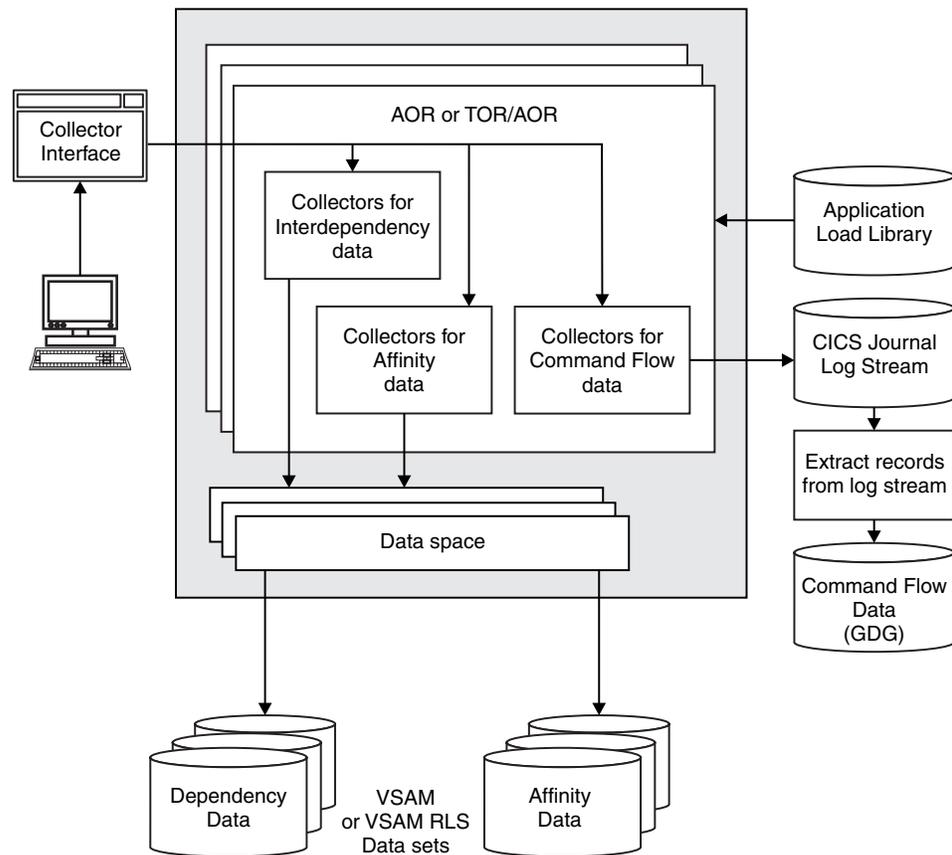


Figure 1. The Collector structure of CICS IA components

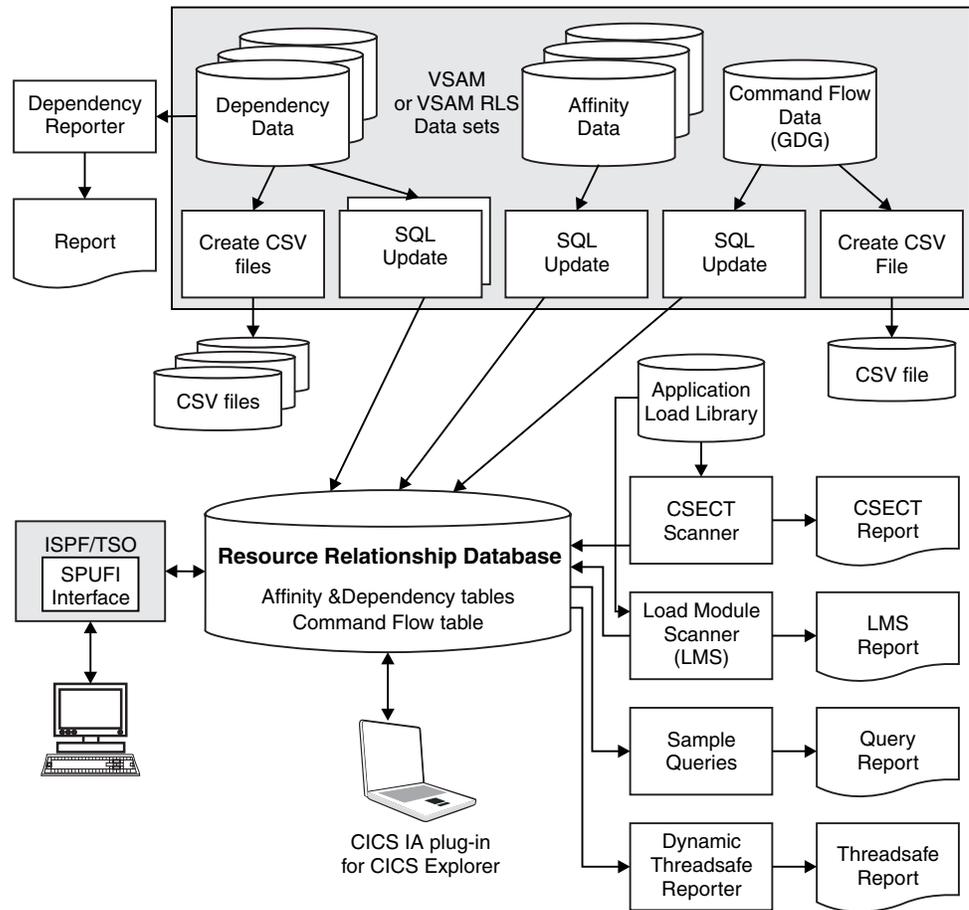


Figure 2. The reporting structure of CICS IA dependency-related components

CICS IA contains these dependency-related components:

The Collector

The Collector is a CICS transaction that runs in your CICS region and intercepts selected CICS and non-CICS programming commands. Depending on what you have specified, it records, in an MVS data space, details of either of the following:

- The resources used by the commands
- The potential affinities created by the commands

You can collect both dependency data and affinity data on the same region at the same time. The dependency data, affinity data, or both are saved to VSAM files.

The Dependency database objects

The Dependency database objects contain data extracted from the VSAM dependency file created by the Collector. It is updated periodically to add data from new or infrequently run applications.

The CICS IA plug-in for CICS Explorer

The CICS IA plug-in for CICS Explorer provides a graphical front end to CICS IA. For more information, see the *Analyzing CICS IA data using the CICS IA plug-in for CICS Explorer* section in the *IBM CICS Explorer User Guide*.

The Dependency Reporter

The Dependency reporter is a batch utility that you can use to convert the dependency data in the VSAM files into reports in a readable format. You might use this function if, for example, you do not have DB2.

The Load Module Scanner

The Load Module Scanner is a batch utility that scans a load module library to detect those programs in the library that issue commands that might cause either of the following:

- Transaction resource dependencies
- Transaction affinities

It produces a printed report. The dependencies data that it collects is written to the Load Module Scanner database objects.

Dependency-related commands

All the commands listed in this section are the dependency-related commands detected by CICS IA.

The dependency-related commands are divided into CICS and non-CICS commands, which are capable of causing resource dependencies, although they might not do so.

For details about CICS and non-CICS dependency related commands, see “Commands monitored for potential dependencies” on page 179.

CICS IA affinity related functions

The affinity related functions of CICS IA help users of CICS dynamic routing, who need to determine whether any of the transactions in their CICS applications use programming techniques that require them to be run in the same region thus creating an inter-transaction affinity, or in a particular region, thus creating a transaction-system affinity. Application programmers can use CICS IA to detect whether the programs they are developing are likely to cause transaction affinities.

The affinity-related functions of CICS IA work in a similar way to the interdependency functions, by collecting information about programs and transactions that issue specific commands, but in this case the objective is to detect affinities rather than interdependencies.

CICS IA detects possible affinities by monitoring those EXEC CICS commands that have the potential to create them. All the CICS API and SPI commands that might create affinities and can be monitored are listed in “Affinity-related commands” on page 12.

The Collector component of CICS IA collects the affinities that apply to a single CICS region, that is, a single application-owning region (AOR) or a single, combined, routing region and AOR. It can be run against production CICS regions and is also useful in a test environment, to monitor possible affinities introduced by new or changed application suites or packages.

The CINT transaction provides an interactive interface with which to control the Collector.

Note: To ensure that you monitor as many potential affinities as possible, run the Collector against all parts of your workload, including rarely used transactions and abnormal situations.

The affinity data collected by the Collector is stored in data tables in a data space. When you stop the Collector and, optionally, at predetermined intervals, the affinity data in the data space is saved to VSAM files by the CICS IA autosave transaction, CINB.

Using CICS IA, you can:

- Collect data about potential affinities
- Load the affinity data into DB2 databases
- Use the Query interface to analyze the affinities data by means of SQL queries
- Use the Load Module Scanner to check a load module library for programs that issue commands that might cause transaction affinities
- Use the Affinities Reporter to produce detailed affinity reports
- Use the Builder to create a file of affinity-transaction-group definitions suitable for input to CICSplex[®] SM

The rest of this section contains:

- “What are transaction affinities?”
- “CICS IA affinity-related components: overview” on page 10
- “Affinity-related commands” on page 12

What are transaction affinities?

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

There are two categories of affinity, inter transaction affinity; see “Inter transaction affinity” and transaction system affinity; see “Transaction system affinity” on page 8.

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 that no transaction affinity exists, indicating no restriction in the choice of available target regions. However, even when transaction affinities do exist, limits to the scope of these affinities are determined by the affinity relations; see “Affinity relations” on page 8 and Affinity lifetime; see “Affinity lifetimes” on page 9.

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, you must review your application to determine whether or not it is suitable for dynamic routing.

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

Transaction system affinity

A transaction system affinity is an affinity between a transaction and a particular CICS region, it is not an affinity between transactions. 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.

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 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. Many different user IDs and terminals associated with the transactions might be included in this affinity group.

LINK3270

All instances of all transactions in the group are associated with the same Link3270 bridge facility.

LUnicode

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 LUnicode relations can be “pseudoconversation”, “logon”, “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”, “sign-on”, “system”, or “permanent”.

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 shared by transactions that take part in the affinity is not recoverable across CICS restarts.

Permanent

The affinity extends across all CICS restarts. The resource shared by transactions that take part in the affinity is recoverable across CICS restarts. This affinity is the most restrictive of all 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 is deleted.

Pseudoconversation

The LUsername or user ID affinity lasts for the whole pseudoconversation and ends when the pseudoconversation ends at the terminal.

Logon

The LUsername 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.

Note:

1. For user ID affinities, the “pseudoconversation” and “sign-on” lifetimes are possible only in those situations in which one user per user ID is permitted. Such lifetimes are meaningless if multiple users are permitted to be signed on with the same user ID at the same time even at different terminals.
2. If an affinity is both “userid” and “LUsername” that is, all instances of all transactions in the group were initiated from the same terminal and by the same user ID, “LUsername” takes precedence.

Worsening of transaction affinities relations:

The worsening of transaction affinities relations is flagged by the Detector and reported by the Reporter.

In some cases, the Detector may 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 Detector records the (worsened) affinity relation as GLOBAL instead of LUNAME or USERID.

Worsening of transaction affinities lifetimes:

Lifetime worsening is flagged by the Detector, and reported by the Reporter.

If a pseudoconversation ends, and the resource still exists, the Detector deduces that the lifetime is longer than PCONV, that is, one of LOGON, SIGNON, SYSTEM, or PERMANENT.

If a logoff or sign-off occurs, and the resource still exists, the Detector deduces that the lifetime is longer than LOGON or SIGNON: that is, 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 system.

In some cases, the Detector may not detect a logoff or sign-off, so cannot be sure that the affinity lifetime is LOGON or SIGNON. In such cases, the Detector records the (worsened) lifetime as SYSTEM or PERMANENT instead of LOGON or SIGNON. For example, this occurs when the CICS region that the Detector is running on is a target region, because it is impossible in some cases to detect a logoff or sign-off that occurs in a connected routing region.

CICS IA affinity-related components: overview

Study the figures to help you understand how the affinity-related components of CICS IA relate to each other.

Refer to Figure 1 on page 4 for the Collector structure.

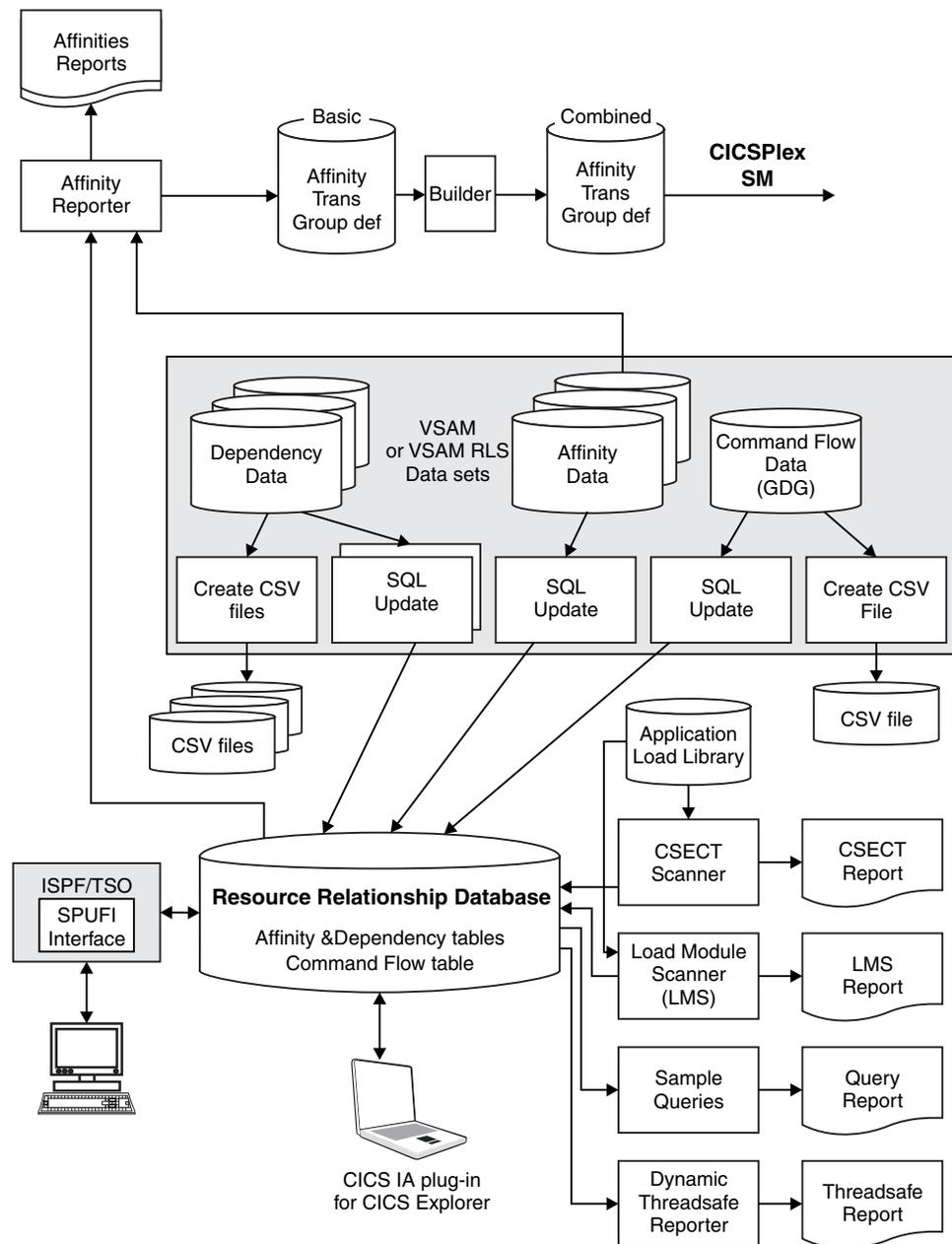


Figure 3. The reporting structure of the CICS IA affinity-related components

CICS IA includes these affinity-related components:

The Collector

The Collector is a CICS transaction that runs in your CICS region and intercepts selected CICS and non-CICS programming commands. Depending on what you have specified, it records, in an MVS data space, details of either of the following:

- The potential affinities created by the commands
- The resources used by the commands

You can collect both dependency data and affinity data on the same region at the same time. The dependency data, affinity data, or both are saved to VSAM files.

The Affinity database objects

The Affinity database objects contain data extracted from the VSAM affinity files created by the Collector. It is updated periodically to add data from new or infrequently run applications.

The CICS IA plug-in for CICS Explorer

The CICS IA plug-in provides a graphical front end to CICS IA. For more information about the CICS IA plug-in, see the *Analyzing CICS IA data using the CICS IA plug-in for CICS Explorer* section in the *IBM CICS Explorer User Guide*.

The Affinities Reporter

The Affinities Reporter is a batch utility that you can use to do any of the following:

- Convert the affinity data in the Affinity database objects into reports in a readable format.
- Convert the affinity data in the VSAM files into reports in a readable format. You might use this function if, for example, you do not have DB2.
- From the affinity data, in the Affinity database objects, 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 Builder

The Builder is a batch utility that takes as input the file of basic affinity-transaction-group definitions created by the Affinities Reporter. It produces a file of “combined” affinity-transaction-group definitions suitable for input to CICSplex SM, which requires that a specific CICS transaction ID (TRANSID) is in only one transaction group.

Affinity-related commands

This section lists the affinity-related EXEC CICS commands detected by the Collector and the Load Module Scanner. All commands listed here are *capable of* causing affinities; they might or might not actually do so.

In Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner:

- The left-hand 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-hand column shows the CICS *SPI* commands that might create transaction system affinities.

Table 1. Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner

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 CSD

Notes:

1. The CICS IA Load Module Scanner might detect some instances of these commands that do not cause an affinity. For example, all FREEMAIN commands are detected but only those used to free GETMAIN SHARED storage might cause an affinity.
2. The CICS IA Load Module 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.
3. The CICS IA 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, causing no affinity problem.
4. The Collector ignores commands that target remote resources and are function-shipped, because function-shipped commands do not cause affinity problems.
5. The Collector and the CICS IA Load Module Scanner do not search for commands issued by any program named CAUxxxxx, CIUxxxxx, or DFHxxxxx, because CICS programs are not considered part of the workload. Also, the Collector does not search for commands issued from:
 - DB2 and DBCTL task-related user exits
 - User-replaceable programs
6. There are other ways in which transactions can cause affinity with each other, but they are not readily detectable by the Collector because they do not take place through the EXEC CICS API.
7. The 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.
8. The Collector and the CICS IA Affinities Reporter ignore ENQ and DEQ commands that specify an ENQSCOPE name.

For details about affinity-related commands see “Commands monitored for potential affinities” on page 192.

CICS IA Command Flow functions

The CICS IA Command Flow utility allows individual users to capture CICS, DB2, MQ and IMS commands in a chronological order for one or more transactions. Each user can capture information for his or her given transaction or transactions. They can also individually load and view the data that they have captured.

CICS IA Command Flow components overview

CICS IA comprises a number of the Command Flow components, which divide into collecting and reporting parts.

CICS IA contains these Command Flow components:

Refer to Figure 1 on page 4 for the Collector structure.

The Command Flow Collector

The Command Flow Collector collects all CICS, DB2, MQ and IMS commands in chronological order and writes them to the CICS Journal log stream.

The Command Flow database objects

The Command Flow database objects contain data extracted from the CICS Journal log stream created by the Collector. It is updated periodically to add data from new or infrequently run applications. See “The structure of the Command Flow table objects” on page 235.

The CICS IA plug-in for CICS Explorer

The CICS IA plug-in for CICS Explorer provides a graphical front end to CICS IA. For more information, see the *Analyzing CICS IA data using the CICS IA plug-in for CICS Explorer* section in the *IBM CICS Explorer User Guide*.

The Command Flow Feature

The Command Flow feature enables you to capture all EXEC CICS, SQL, MQ and IMS calls in chronological order.

With the Command Flow Feature you can trace the command flow in up to five transactions in chronological order. A trace name can be associated with each instance of the trace. CICS IA uses a number of CICS Global User Exits (GLUEs) and a CICS Task Related User Exit (TRUE) to intercept commands. The command records are written to a CICS User Journal, which uses the MVS logger subsystem to write them to a log streams data set. At the end of a trace, a record containing the name, start time, end time, and the five possible transactions is written to the journal.

The data is read from the log stream data sets into a generation data set. The data in the generation data set is formatted to update the CIU_CMDFLOW_DATA and CIU_CMDFLOW_INDEX DB2 tables, or to create QSAM data sets with the data stored in the Comma Separated Value (CSV) format. See Figure 4 on page 15.

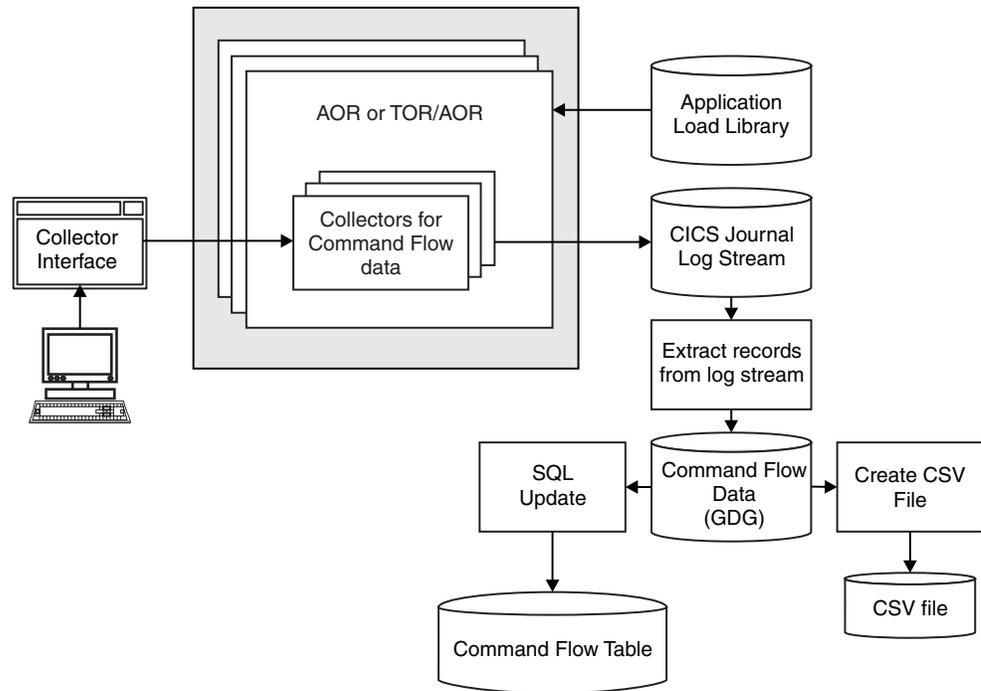


Figure 4. Command Flow option structure

The Collector component

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

You can collect both dependency data and affinity data on the same region at the same time.

The Collector saves details of the dependencies or affinities in an MVS data space. This data is subsequently saved to storage. The details of the Command flow are saved in the CICS Journal Log Stream. The Collector consists of:

- A control transaction, CINT
- An autosave transaction, CINB
- A control transaction for Command Flow data collection, CINC
- Some global user exit programs
- A task-related user exit program

The Collector components are shown in Figure 5 on page 16.

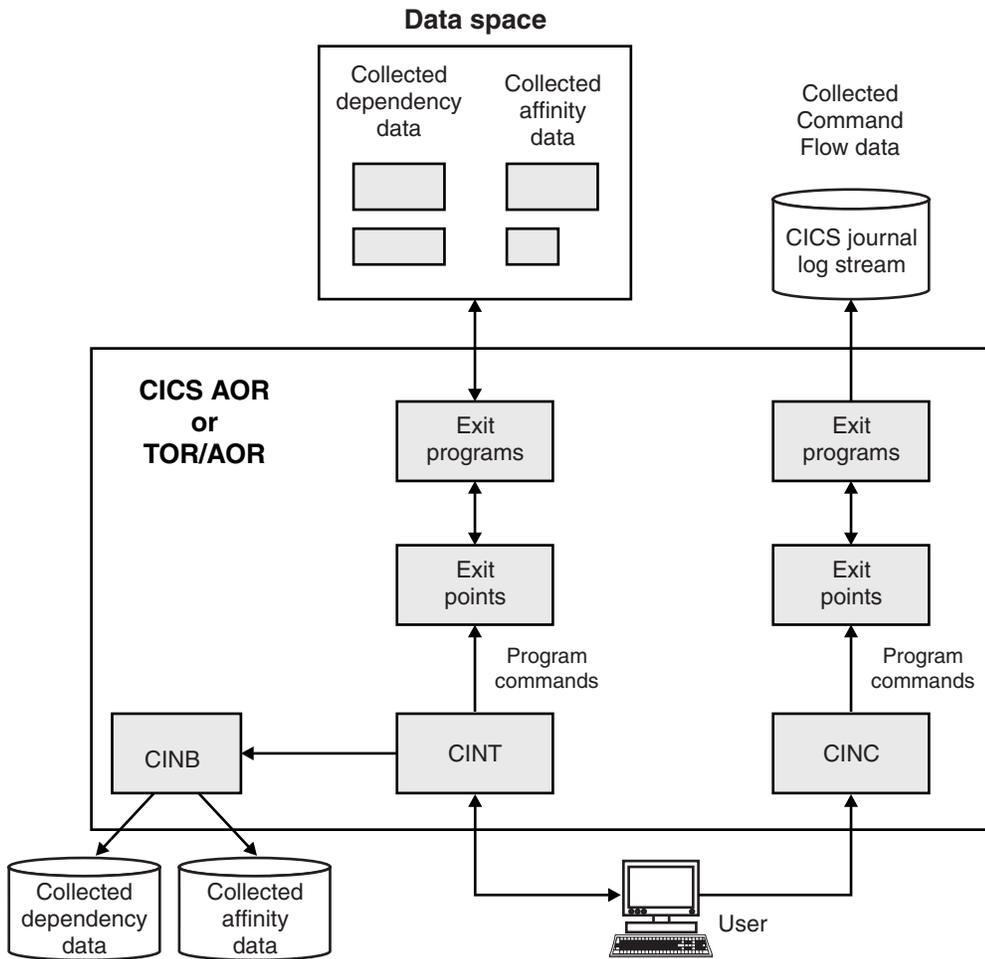


Figure 5. Collector components

Dependency data is collected by global user exit programs at the exit points shown in Global user exits used by CICS IA to collect resource dependencies. Affinity data is collected by the task-related and global user exit programs shown in Global and task-related user exits used by CICS IA to collect transaction affinities. Command Flow data is collected by the task-related and global user exit programs shown in Global and task-related user exits used by CICS IA to collect transaction affinities. The exit programs coexist with any other exit programs that are invoked at the same exit points.

You are recommended to place the CICS IA exit programs after any other exit programs that are invoked at the same exit points and make them the last to be enabled. This enables CICS to capture, where applicable, the correct remote SYSID associated with resources such as programs, files, TS queues, TD queues, and transactions.

Where more than one global user exit program is invoked from the same global user exit point, the order in which the programs are invoked is the order in which they are activated by EXEC CICS ENABLE commands. For more information, see "Invoking more than one exit program at a single exit" in the *CICS Customization Guide*.

Run the Collector in stable CICS regions only. Do not apply maintenance to application programs while the Collector is running. Such maintenance could introduce or remove dependencies or affinities, thus rendering collected data inaccurate.

What can be monitored

The commands that the Collector actually monitors depend on the way you configure it.

See “Controlling the Collector” on page 18 and “What is not monitored” for more information on the configuration of the Collector. The Collector can monitor:

- The EXEC CICS API commands, listed in Dependency-related CICS API commands detected by the CICS Interdependency Analyzer, that can cause transaction dependencies
- The EXEC CICS SPI commands, listed in Dependency-related CICS SPI commands detected by the CICS Interdependency Analyzer, that can cause transaction dependencies
- The CICS Front End Programming Interface (FEPI) API commands, listed in Dependency-related CICS SPI commands detected by the CICS Interdependency Analyzer, that can cause transaction dependencies
- The CICS FEPI SPI commands, listed in Dependency-related CICS FEPI SPI commands detected by the CICS Interdependency Analyzer, that can cause transaction dependencies
- The non-CICS API commands, listed in “Non-CICS API commands detected” on page 191, that can cause transaction dependencies
- The EXEC CICS API and SPI commands, listed in Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner, that can cause transaction affinities
- The dynamic COBOL calls, collected by the Dependency and Command Flow Data collectors for the LE COBOL programs if the Language Environment option, CBLPSHPOP, is active and the load program name for the calling and the called programs correspond to the program name in the LE prolog.

For information about the Language Environment runtime CBLPSHPOP option refer to *the CICS TS Application Programming Guide*, section *Language Environment CBLPSHPOP option*.

For information about what the Collector reports for each monitored command, see Appendix A, “Details of dependencies and affinities collected,” on page 179.

As well as monitoring program commands, the Collector also collects information about the CICS regions on which it runs and stores that information in the CIU_REGION_INFO database table. For example, for each collection of dependency or affinity data, the table contains the names of the CICS System Definition data set (CSD) and the first four resource group lists in the CSD.

What is not monitored

The Collector does not monitor all program commands.

The Collector does not monitor program commands when:

- The Collector is not running.
- The issuing program was translated with the SYSEIB option. An exception here is the EXEC CICS PUSH HANDLE command. Monitoring these commands enables CICS IA to capture dynamic COBOL calls.

- You are collecting interdependency data and the command is not one that can cause transaction resource dependency.
- You are collecting affinity data and the command is not one that can cause transaction affinities.
- The program is a DB2 or DBCTL task-related user exit.
- The program is a CICS user-replaceable program.
- The program issues a CALL to named COUNTERS.
- The transaction identifier does not match the prefix, if specified, for transactions to be monitored. See “Specifying region-specific options: general” on page 93.
- The command is not in the relevant subset of command-types specified to be monitored. There are three such subsets: one for dependency-related CICS commands, one for dependency-related DB2, IMS, and MQ commands, and one for affinity-related CICS commands: see “Specifying region-specific options: API and SPI commands to be monitored” on page 98, “Specifying which dependency-related DB2, IMS, MQ commands and TRUES are to be monitored” on page 100, and “Specifying which affinity-related CICS commands are to be monitored” on page 102.
- The first few letters of the program name match a prefix in the list of “excluded” program prefixes, which identify programs for which data is not to be collected. See “Creating a program exclude list” on page 55.
The default program exclude list excludes programs with names that start with “ABL”, “CAU”, “CBM”, “CEE”, “CIU”, “CME”, “CPA”, “CSQ”, “DFH”, “DSN2”, “DSNC”, “DWW”, “EDC”, “EQA”, “EYU”, “IBM”, “IN25”, “IGZ”, “ISZ”, or “VID”.
- The first few letters of the transaction name match a prefix in the list of “excluded” transaction prefixes, which identify transactions for which data is not to be collected: see “Creating a transaction exclude list” on page 56.
- The command causes an un-handled abend.

The Collector does not monitor pseudoconversations in which you continue a pseudoconversation by setting a transid in the TIOA rather than by using RETURN TRANSID.

Ideally, CICS IA will ignore commands issued by task related user exits and global user exits because they are not part of applications. However, it cannot distinguish such commands from others, and does monitor them. If your user exits use commands that can cause transaction dependencies, the commands are monitored, perhaps making any dependency problem seem worse than it actually is.

When capturing dynamic COBOL call, the Collector can record the name of the enclave program and an undefined command offset under one of the following conditions:

- The called program does not conform to the LE prolog and LE linkage conventions.
- The ADABAS or Natural program request is received.
- The DBCTL request that relates to the recorded EXEC DLI command is received.

Controlling the Collector

You can monitor and control the Collector through the CINT transaction. Also you can control the Collector from CICS Explorer through the CICS IA plug-in for CICS Explorer.

For example, the CINT transaction enables you to:

- Start, pause, continue, and stop the collection of dependency or affinity data.
If necessary, you can start, stop, pause, or resume the collection of data on multiple regions at once, that is, with a single CINT command. How to do this is described in “Controlling the collection of dependency and affinity data” on page 80.
You can also set timers, one per region, to control the dates and times at which dependency or affinity data is collected. A timer pauses and resumes the Collector automatically at predetermined times. How to set a timer is described in “Specifying region-specific options: timers” on page 103.
- Specify which type of data, dependency or affinity, is to be collected on each region.
- Specify for which programming commands, and for which transactions, data is to be collected. For example, you could specify that only dependency data for transactions with names beginning with “PAY” will be collected; and that, within this subset of transactions, only EXEC CICS file control commands will be monitored.
- Specify, by means of a list of name prefixes, a set of programs for which data is not to be collected. See “Specifying region-specific options: general” on page 93.
- Specify, by means of a list of name prefixes, a set of CICS transactions for which data is not to be collected.
- Set default values for the Collector's region specific options. How to set default values is described in “Changing the Collector options” on page 90.
- Set a region-specific Collector option to the same value on multiple regions with a single CINT command. How to set this option is described in “Changing the Collector options” on page 90.
- Specify, by means of the USER-control record in the CINT control file, individual user's Command Flow options and region configurations.
- Control USER-records (ADD, COPY, DELETE and DETAIL).

For detailed information about controlling the Collector, see Chapter 5, “Running the Collector,” on page 71.

The options that you specify to control the Collector for a CICS region are preserved in a recoverable VSAM control file. For more information about this file, see “The control record VSAM file” on page 22.

How dependency data is collected

The Collector uses tables in a data space to hold collected dependency data.

These tables are the main dependency-related tables:

- The CICS table, which records every dependency on a CICS resource.
- The DB2 table, which records every dependency on a DB2 resource.
- The IMS table, which records every dependency on an IMS resource.
- The MQ table, which records every dependency on an MQ resource.
- The Natural table, which records every dependency on a Natural resource.
- The DTP table, which records each ALLOCATE where the Collector has not found a matching SEND, FREE, CONVERSE, or CONNECT PROCESS for the same convid or session.

- The MQX table, which records the MQ queue name used on each MQOPEN command. The entry is used on subsequent MQPUT and MQGET commands for the same task. It is removed at MQCLOSE.
- The file table, which records detailed file information.
- The program table, which records detailed program information.
- The transaction table, which records detailed transaction information.
- The TDQueue table, which records detailed TDQueue information.
- The TSQueue table, which records detailed TSQueue information.
- The exit data table, which records detailed exit information.
- The Web service data table, which records detailed Web service information.

The dependency tables reside in the data space. The tables, excluding the DTP and MQX tables, are saved to VSAM files when you stop the Collector and, optionally, at predetermined intervals.

How affinity data is collected

The Collector uses tables in a data space to hold collected affinity data.

The main affinity-related tables are of the following types:

- The affinity group table, which records every affinity-transaction-group; that is, every group of CICS transactions that have been grouped together because they have the potential to create the same type of affinities.
- The affinity command table, which records every unique combination of:
 - EXEC CICS command with the potential to create an affinity
 - Program
 - Transaction ID

The affinity tables reside in the data space. The affinity group and affinity command tables are saved to VSAM files when you stop the Collector and, optionally, at predetermined intervals.

Saving data

The dependency and affinity data collected by the Collector is saved to CICS IA VSAM files by the autosave transaction, CINB.

For more information about these files, see “The dependency data and affinity data VSAM files” on page 21.

The CINB transaction is invoked automatically by CICS; it is not a user transaction. The CINB transaction is invoked in a number of ways:

- When you pause or stop the Collector.
- If the **Periodic Save** option is selected, every five minutes.
- When the trigger value is reached. You can define the value “n” in thousands of updates to the data space. The value for “n” can be in the range 2 to 9999. A value of “1” will not trigger the start of the CINB transaction.

The CINB transaction saves dependency and affinity data automatically when you stop the Collector. You can also specify that data is saved as follows:

- On a predetermined time and activity basis. That is, data is saved if either more than 300 seconds has passed or more than 1 000 table elements have changed, since the last save.
- When you pause the Collector.

After the CINB transaction has saved any data collected, it either becomes dormant until next activated while the Collector is still running, or pauses or terminates if the Collector has been stopped.

Only the CICS, DB2, IMS, MQ, affinity, and resource detail tables in the data space need to be saved. The DTP and MQX tables are not saved because they hold only temporary data about DTP conversations and MQ queue names. Also, when data is saved, only those table elements that have been added or changed since the last save are written to the file. Time stamps in each table element indicate whether the element has been written already, and whether it has changed since the last write, to minimize the number of writes performed.

The dependency data and affinity data VSAM files

To control the operation of multiple instances of the Collector, running on different CICS regions, from a single CICS terminal, you must share the dependency data files, the affinity data files, and the control record file across all the regions being monitored.

To do so, you can use either of these methods:

- VSAM record-level sharing (RLS). For information about using VSAM RLS in CICS, see the *CICS Installation Guide*.
- Function shipping to a file-owning region (FOR). For information about CICS function shipping, see the *CICS Intercommunication Guide*.

The alternative is to define local dependency and affinity data files, and a control record file, on each region to be monitored. In this way, from a CICS terminal you can control only an instance of the Collector running on the local region.

The Collector uses a separate, nonrecoverable, VSAM KSDS files to record dependencies on each of the following:

- CICS resources with names up to 32 bytes long: file CIUINT1
- DB2 resources: file CIUINT2
- IMS resources: file CIUINT3
- MQ resources: file CIUINT4
- CICS resources with names longer than 32 bytes: file CIUINT5
- Detail data resources: file CIUINT6
- Natural resources: file CIUINT7

Ensure that each file is big enough to hold the maximum amount of dependency data that might be collected.

The Collector uses other nonrecoverable VSAM KSDS files to record intertransaction and transaction-system affinities:

- Affinity resources with a 16-byte key: file CIUAFF1
- Affinity resources with a 32-byte key: file CIUAFF2
- Affinity resources with a 224-byte key: file CIUAFF3

Ensure that each file is big enough to hold the maximum amount of affinity data that might be collected.

KSDS files are used because the Collector and the Dependency and Affinity Reporters need keyed access to the data. The files are not recoverable because of the large amount of data that might need to be written.

The dependency data files and affinity data files contain a header record for each CICS region that has been or is being monitored by the Collector. The header

record enables both the Collector and the Reporter to validate that the files that are presented are data files suitable for CICS IA. The header record has a key in the same format as the rest of the keys on the file, so a table identifier of zero is used. No real table will have a table identifier of zero. The header record contains the CICS specific APPLID, thus allowing files to be cross validated.

The control record VSAM file

The CICS IA control file is a recoverable VSAM KSDS file that holds a single header record that holds global options that apply to all the CICS regions, a single DEFAULT control record that contains the default values for all regions and one control record for each CICS region being monitored that contains options that overrides the default record.

- A single header record that holds global options that apply to all the CICS regions that have been or are being monitored by the Collector. How to specify the Collector global options is described in “Changing global options” on page 114.
- One DEFAULT control record for all CICS regions. The control record contains region-specific options and statistics that are maintained across Collector runs, transaction failures, system failures, and restarts.

Each control record holds the following, region-specific, information:

- CINT options for this region. How to specify the Collector region-specific options is described in “Specifying region-specific options: region configuration” on page 90.
- The APPLID and SYSID of the CICS region.
- Collector statistics.
- History information:
 - Reason why STOPPED
 - user ID if STOPPED by user
 - Abend code if STOPPED by abend
 - user ID for last Collector options update
 - Date and time of last Collector options update

The record is updated whenever any of the above information changes, when the Collector options or statistics for this region change, or, the Collector state changes to STOPPED on this region.

- One control record for each CICS region that has been or is being monitored by the Collector. The control record contains region-specific options and statistics that override the options set by the DEFAULT record.

USER control record

The CICS IA control file also holds a USER control record that is used by the CINC transaction and contains individual user's Command Flow options and regions configurations.

The CICS IA User Command Flow Utility uses the control file for regions configuration data, global options and user's control data. The administration functions for these types of information are supported by the CINT transaction, which supplies options to add, copy, delete and detail user's control data. The initial values of user's control data are default values.

To control the operation of multiple instances of the Command Flow Utility, running on different CICS regions, from a single CICS terminal, it shares the

Control record file across all the regions being monitored. To do so, you can use either a VSAM record-level sharing (RLS) or a function shipping to a file-owning region.

The Dependency database objects

The Dependency database objects contain accumulated data about all your programs and transactions and the resources that they use.

You also have the option to group your transactions into applications so that you can query application dependencies. Update the database regularly to add new information recorded by the Collector in the VSAM dependency files. CICS IA provides a job to create the database objects and a suite of batch programs to update the database. See “Updating the Dependency database objects” on page 121 for more details.

Typically, there is only one set of Dependency database objects, even if you have separate dependency data and control record files for each region monitored by the Collector. In the latter case, you would typically feed the information in every VSAM dependency data file into the one set of Dependency database objects. Using one database makes it easier to compare and contrast dependency data from different regions.

For details about the dependency database objects, see Appendix C, “The structure of the CICS IA database,” on page 203.

In addition facilitating tables are used in the updates of the base tables.

The Affinity database objects

The Affinity database objects contain accumulated data about all your programs and transactions and the affinities between them.

You can also, optionally, group your transactions into applications so that you can query application affinities. Update the database objects regularly to add new information recorded by the Collector in the VSAM affinity files. CICS IA provides a job to create the database and a suite of batch programs to update the database. See “Updating the Affinity database objects” on page 122 for more details.

Typically, there is only one set of Affinity database objects, even if you have separate affinity data and control record files for each region being monitored. In the latter case, you would typically feed the information in every VSAM affinity data file into the one set of Affinity database objects. Using one database makes it easier to compare and contrast affinity data from different regions.

For the affinity base tables see “Affinity base tables” on page 211.

In addition, facilitating tables are used in the updates of the base tables.

The CICS IA plug-in for CICS Explorer

The CICS IA plug-in for CICS Explorer (CICS IA plug-in) is an Eclipse plug-in that operates on top of the IBM CICS Explorer to help you analyze the CICS IA data, including the interdependency data, affinity data and Command Flow data.

Using the CICS IA plug-in, you can perform the following tasks:

- For the dependency data you can:
 - View resources used by a region, transaction or program.
 - View the programs or transactions that use a given resource.
 - View resources used by an application.
 - Create your own queries to analyze the data.
 - View detailed information for programs, transactions, files, TSQueues, Events, Exits or Regions.
 - Compare resources used by a transaction, program or region.
- For the affinity data you can:
 - View Affinities by Region.
 - View Affinities by Transaction.
 - View Affinities by Program.
- For the Command Flow data you can:
 - View collections by time or userid.
 - View tasks within a Collection.
 - View the execution tree for a given task.

You can also use the CICS Explorer to operate the controller. You can perform the following tasks:

- For a given CICS region you can issue a START, STOP, PAUSE, CONTINUE or REFRESH of the dependency or affinity collection.
- For a given user you can START or STOP the Command Flow feature.
- For more information about the IBM CICS Explorer, see <http://www.ibm.com/cics/explorer>.

CICS IA reports

CICS IA can create dependency reports, affinity reports and threadsafe reports by running batch jobs.

- The Dependency Reporter.
- The Affinities Reporter.
- The Threadsafe Reporter.

The Dependency Reporter

The Dependency Reporter consists of a batch job that converts the dependency data collected by the Collector into reports that present the data in a readable format.

The files of dependency data produced by the Collector are the input to the Dependency Reporter. Depending on how you configure the Reporter job, the output might be, for example, a listing of the CICS, DB2, MQ, or IMS commands that were monitored, naming the transactions and programs where they occurred, the resource being acted upon, and other details.

If the dependency data files are shared by multiple regions, you can run one Dependency Reporter job to produce a report showing dependencies, for example, dependencies on DB2 resources, found in either of the following:

- A single, specified, region
- All of the regions

If, however, each monitored region has its own, region-specific, dependency data files, each Dependency Reporter job always retrieves data for a single region; to retrieve data from multiple regions, you must run your job multiple times, against the relevant dependency files for each region in which you are interested.

The Affinities Reporter

The Affinities Reporter consists of a batch job that converts the affinity data collected by the Collector into reports presenting the data in a readable format. It can also be used to create a file of affinity-transaction-group definitions in a syntax approximating the batch API of CICSplex SM. This file is used as input to the Builder component.

The files of affinity data produced by the Collector are the input to the Affinities Reporter. Depending on how you configure the Affinities Reporter job, the output might be, for example:

- A listing of possible transaction-system affinities for a particular CICS region, naming the transactions and programs involved, and the affinity relations and lifetimes
- A file of affinity-transaction-group definitions

If the affinity data files are shared by multiple regions, you can run one Reporter job to produce a report showing affinities, for example, inter-transaction affinities, found in either of the following:

- A single, specified, region
- All of the regions

If, however, each monitored region has its own, region-specific, affinity data files, each Affinities Reporter job always retrieves data for a single region; to retrieve data from multiple regions, you must run your job multiple times, against the relevant affinity files for each region in which you are interested.

The Threadsafe Reporter

The Threadsafe Reporter consists of a batch job that produces reports displaying the threadsafe status of each command in the requested programs.

The threadsafe status for a command can be as follows:

Threadsafe

An EXEC CICS command that does not cause a TCB swap.

Non-Threadsafe

An EXEC CICS command that can cause a TCB swap.

Indeterminate Threadsafe

An EXEC CICS command where it cannot be determined if the call causes a TCB swap.

Dynamic call

A call to another module a execution time. The call was not initiated using an EXEC CICS command.

Threadsafe Inhibitor call

An EXEC CICS command that can cause an unsafe affinity between transactions. The call needs to be investigated before knowing if it inhibits the program from being threadsafe. These commands are ADDRESS CWA, LOAD HOLD, GETMAIN SHARED, and EXTRACT EXIT.

DB2 calls

The calls to the CICS DB2 interface are threadsafe.

IMS calls

The calls to the CICS IMS interface are non-threadsafe.

MQ calls

The calls to the CICS MQ interface are threadsafe only in CICS TS V3.2.

To request a Dynamic Analysis Threadsafe report, edit and run the CIUJTSQ2 job.

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

The Scanner component

The Scanner component consists of two scanners: the Load Module Scanner and the CSECT Scanner.

The Load Module Scanner

The Load Module Scanner is a batch utility that scans a load module library to detect those programs in the library that issue commands that might cause transaction dependency or transaction affinities.

For EXEC CICS commands, the Load Module Scanner examines the individual object programs looking for patterns matching the argument zero format for such commands. 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 two bytes of this constant identify the command; for example, X'0A04' identifies it as a READQ TS command.

The Load Module Scanner:

- Detects the use of:
 - The dependency-related commands listed in “Dependency-related commands” on page 6
 - The affinity-related EXEC CICS API and SPI commands 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
- Produces a printed report
- Writes the affinity-related data that it collects to the Load Module Scanner database objects

The report produced by the Load Module Scanner indicates only that potential dependency or affinity problems might exist because it only identifies the programs that issue the commands. It cannot obtain dynamic information about

the transactions using the programs or the names of the resources acted upon. Use the report in conjunction with the main reports produced by the Dependency and Affinity Reporters. See “The Dependency Reporter” on page 24 and “The Affinities Reporter” on page 25.

Note:

1. The Load Module Scanner operation is independent of the language that the scanned program was written in and the release of CICS the scanned program was translated under.
2. The Load Module Scanner might indicate a dependency or affinity problem that does not really exist, because the bit pattern found accidentally matches the argument zero format for a dependency command.

The Load Module Scanner database objects

The Load Module Scanner database objects contain accumulated data, collected by the Load Module Scanner component, about programs and commands that might cause affinities. The purpose of the set of Load Module Scanner database objects is to allow you to compare, using SQL commands, the data produced by the Load Module Scanner to that produced by the Collector.

The CSECT Scanner

The CSECT Scanner scans load modules for information that can be used to identify the version of each CSECT.

The output is stored in DB2 tables and can be used, in conjunction with the DB2 dependency tables, to identify different versions of programs. For information about using the CSECT Scanner, see Chapter 11, “Running the CSECT Scanner,” on page 153.

The Builder component

The Builder is a batch utility that takes as input a file of basic affinity-transaction-group definitions created by the Reporter. It produces a file of “combined” affinity-transaction-group definitions suitable for input to CICSplex SM.

You must combine the basic transaction groups because of a CICSplex SM rule stating that a specific CICS transaction ID (TRANSID) can appear in only one transaction group. Because a TRANSID might appear in more than one basic group, you must combine them to form larger groups to satisfy CICSplex SM.

Chapter 2. Getting started with CICS IA

In this section, you can read basic information about the CICS IA components and their usage so that you can start working right away. Some sections also contain links to topics that cover specific subjects in more detail.

Running the configuration utility

The SMP/E installation process for CICS IA is described in the Program Directory that is distributed with the product. The first task to complete after the SMP/E install is to run the configuration ISPF utility.

The installation and customization program helps you to customize the CICS IA sample jobs, lists, and SQL definitions. It creates customized installation jobs in which the names of system entities, such as the high-level qualifier (hlq) of the CICS IA data sets, are set to specified values to suit your local environment.

The installation and customization program does not change existing original libraries, it only creates a new set of those.

To invoke the installation and customization program, run member CIUCNFG1 of the SCIUEXEC library using the ISPF command shell:

```
tso ex 'CICSIA.V320.SCIUEXEC(CIUCNFG1)' 'CICSIA.V320 ENU'
```

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

- The high-level qualifier of the CICS IA data sets. In our case it is CICSIA.V320.
- The national language to be used. In our case it is ENU.

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

- CICS IA variables: In this section, specify the high-level qualifier of the CICS IA libraries and the qualifier and size of the CICS IA VSAM files.
- DB2 variables: In this section, provide information about the DB2 subsystem where CICS IA is installed. You might need to contact your DB2 system administrator to get the information about your DB2 subsystem.
- CICS variables: In this section, provide the CICS transaction server information that is required to install CICS IA and support usage of the LIBRARY resource.
- Migration variables: If you did not migrate from a previous version of CICS IA, there is no need to specify any information in this section. However, you can use the migration variables section to provide details about the current and target environment.
- Explorer plug-in variables: In this section, provide information about the resources that are required to invoke Atomservice.
- General variables: In this section, specify which assembler program and language environment qualifier you use.

For the full list of all variables and their descriptions, see Appendix E, “Worksheet for the installation customization program,” on page 317.

Customizing your CICS regions to use the CICS IA Collector

There are several basic steps that enable you to start using CICS Interdependency Analyzer effectively.

Before you start collecting data, configure the target region:

- Create VSAM files to store data
- Create custom CICS resource definitions
- Configure the CICS IA startup job
- Collect DB2 resources

Make sure you restart your CICS region after all these operations, so that all the changes are applied. You can find more detailed descriptions of each procedure in the following tasks.

Creating VSAM files

The dependency and affinity data collected by the Collector is saved to CICS IA VSAM files. You need to create them before running the Collector.

To create CICS IA VSAM files, run jobs CIUJCLCC and CIUJCLCA. When the installation and customization program completes successfully, you can find customized copies of both jobs in the SCIUSMP1 data set.

For running the collector on a single CICS region, use the customized jobs to define non-RLS files. In case you want to run the collector on multiple regions, make sure to share the dependency and affinity data files and the control record file across all the regions. To share the files, you can use VSAM RLS or function shipping of file control requests to a file owning region.

Defining resources to CICS

Create custom resource definitions by editing and running sample jobs available with CICS IA.

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 CINC
- VSAM files CIUCNTL, CIUINT1 through CIUINT7, and CIUAFF1 through CIUAFF3
- DB2 entry (DB2ENTRY) definitions and DB2 transaction (DB2TRAN) definitions
- CINT transient data queue for messages
- Atomservice resources
- LIBRARY resources (available for CICS IA V3.2 and above).

To define the CICS resources for the CICS IA collector, edit and run job CIUJCINT.

Note: In this job, you can also define LIBRARY resources for the SCIULOAD and SCIULODE/K data sets, and they will be loaded dynamically during the startup job execution.

To define the CICS Atomservice resources, edit and run job CIUJWEB2.

To define the CICS file resources for the CICS IA Collector, edit and run one or both of the following jobs:

- CIUJCFIL to define the CICS file resources locally or in a file-owning region
- CIUJCFIR to define the CICS file resources as remote to a file-owning region.

Tailoring your CICS startup job

To enable CICS IA to run in your CICS region, configure your CICS startup job.

About this task

To set up the CICS startup job, 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

Or you can edit the CIUJCINT job to define LIBRARY resources for the SCIULOAD and SCIULODE/K data sets, and they will be loaded dynamically during the startup job execution.

Note: Available for CICS IA V3.2 or above.

5. Add the following DD statement for the CINT transient data message log:

```
//CINTLOG DD SYSOUT=*
```
6. On each region where you intend to collect DB2 data, ensure that the user ID, under which the CINB transaction runs, has permission to access the SYSIBM.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 already has such permission.

Collecting DB2 resources in your CICS region

You must edit the JCL to conform with your own system conventions.

About this task

Before collecting DB2 resource information in your CICS region, it is highly recommended that you create an new index against your SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT tables. This will improve performance of the online collection. If you choose not to create these indexes, switch off the DB2 query in the SYSIBM tables by entering N in the Inquire for DB2 Resources field in the CINT General Options panel for that region. See “Specifying region-specific options: general” on page 93 for details.

To collect DB2 data:

1. Review the following members in hlq.SCIUSMP2.OUT:

- CIUDBCT
 - CIUDBNT
2. Review the associate SQL members for the above jobs. They can be found in hlq.SCIUSQL.OUT.
 3. Run hlq.SCIUSMP2.OUT(CIUDBCT) to create indexes for SYSIBM.
 4. Run hlq.SCIUSMP2.OUT(CIUDBNT) to bind the programs that update DB2 tables.

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.

Customizing DB2 Environment

Besides configuring the CICS Interdependency Analyzer Collector environment itself, you also need to adjust its working DB2 environment.

You can do it in four steps:

- Create the CICS IA database
- Bind the programs that update DB2 tables
- Load some sample interdependency data into the Dependency database objects
- Load the CIU_VERSION table

Creating the CICS IA database

CICS IA provides sample job hlq.SCIUSMP2.OUT(CIUDBNT) that binds all DB2 load programs into a DB2 plan. The final step in the sample job runs the DB2 GRANT command that gives access to users who require to run the DB2 load jobs. The GRANT command can be found in member hlq.SCIUSQL.OUT(CIUGRNTB).

To create a CICS IA database, do the following:

1. Edit and review the sample job CIUDBNT.
2. Edit and review the GRANT command in CIUGRNTB.
3. Submit the sample job CIUDBNT.

Binding the programs that update the DB2 tables

CICS IA provides sample job hlq.SCIUSMP2.OUT(CIUDBCQ) that defines all the CICS IA database objects required to store collected data. These objects include TABLES, INDEXES, VIEWS, and Stored Procedures. The sample job runs the Data Definition Language (DDL) commands required to create these objects. The DDL is stored in several members of hlq.SCIUSQL.OUT.

To perform this task, do the following:

1. Edit and review the sample job CIUDBCQ.
2. Edit and review the sample DDL in hlq.SCIUSQL.OUT(CIUMAIN).
3. Submit the sample job CIUDBCQ.

Running the CIUIVPLD job

CICS IA provides sample CICS data that can be loaded into the CIU_CICS_DATA table. This enables you to use the CICS IA Explorer plug-in to view sample data and verify that CICS IA database is created and the connection to the Explorer is working. CICS IA provides sample job hlq.SCIUSMP2(CIUIVPLD) to load this data into the CIU_CICS_DATA table. The data itself is contained in hlq.SCIUDAT(CIUIVPC).

To load sample data to the CIU_CICS_DATA table, edit and run the CIUIVPLD job.

Loading the CIU_VERSION table

CICS IA uses a DB2 table called CIU_VERSION to control which version of the DB2 tables the CICS IA Explorer plug-in is connected to. CICS IA supplies sample job hlq.SCIUSMP2(CIUVERLD) to populate this table.

To load the CIU_VERSION table, edit and run the CIUVERLD job.

Running the Installation Verification Program

Before starting to use CICS IA, you need to make sure that the installation has completed successfully.

This can be done with the help of the Installation Verification Program (IVP) that checks CICS RDO definitions to ensure that all software elements (programs, maps, transactions, files, TD-Queues, and DB2 entries) are correctly defined and available.

To check that CICS IA is configured and installed correctly to the target CICS TS region, run the CIUT transaction from the region that initiates the IVP. Upon successful completion, the following message is displayed:

```
CIU1009 INSTALLATION VERIFICATION ENDED SUCCESSFULLY
```

Collecting data with CICS IA

The Collector enables you to detect existing resource dependencies in a running CICS region in a real-time mode. With this tool, you can collect data on both types of resource dependencies identified by CICS IA, which are the direct dependencies and affinities.

Collecting and analyzing the affinities data is especially helpful when managing multiregional environments (CICSplex). The Collector detects dependencies between several transactions (inter-transaction affinities) or a transaction and a particular CICS region (transaction-system affinities), enabling you to define the shared resources and, therefore, configure CICS more effectively. For the purpose of enhancing a separate system, collect the dependency data. Run the Collector on a particular region to detect which resources a transaction consumes and to monitor the way it interacts with the system.

Configuring the Collector

You can manage and control the Collector with the CINT transaction, which enables you to start, pause, continue and stop the data collection process.

About this task

To access the configuration options, enter CINT at the CICS terminal, and the Collector Main Administration Menu appears. From this screen, you can go further to more detailed options menus and customize the process to the full extent. When running the Collector for the first time, be sure to specify the region options:

1. On the Collector Main Administration Menu screen, type 2 to open the Region Configuration Menu screen, and press Enter.
2. Specify the region to monitor: on any line, enter the new region's APPLID and SYSID in the New Applid and New Sysid fields, respectively. Type 1 in the Act field and press Enter. CICS IA adds the new region to its list of regions to be monitored, giving system default values to all region-specific Collector options for the new region.
3. Specify the dependency-related DB2, IMS, and MQ commands to be monitored. First, open the Resource Options screen by typing 4 and pressing Enter. Then open the DB2/MQ/IMS Options screen by typing 5 and pressing Enter. Then, go through the list of commands and for each type of command that you want the Collector to monitor, type Y. For each type of command that you don't want the Collector to monitor, type N, or, to use the default value in the CICS IA control file, specify a blank.
4. Set the region specific options, such as which CICS transactions should be monitored and whether dependency or affinity data is to be collected. To do this, go back to the Region Configuration Menu and enter 1 to open the General Options screen. In the Data to Collect field, type A for affinity or I for dependency data collection, or B for both affinity and dependency data collection.

Running the Collector

As soon as the Collector is configured, you can start collecting data.

About this task

The procedure comprises the following steps:

1. Start the Collector. The easiest way to do that is to type CINT START at the terminal and press Enter. This one is a basic command that initiates data collection on the local CICS region, however, you can modify it by adding the options postfixes. For the list of available options and other ways of launching the Collector, see "Starting the collection of data" on page 81.
2. Start the target transaction.
3. Pause or stop the data collection process, when you feel that sufficient time has elapsed. You can do it via the CICS terminal by entering CINT PAUSE or CINT STOP. Just as the CINT START command, these two are very basic and affect only the local region, but you can make them more specific using the options postfixes. For lists of options and alternative ways of stopping the Collector, see "Pausing the collection of data" on page 85 and "Stopping the collection of data" on page 87.

Loading CICS IA data

When the data is collected, you need to load it to the database.

The Collector stores all the accumulated data in VSAM files, and you need to send these updates to your database in order to view the results. CICS IA provides sample jobs for updating both dependency and affinity databases: for the

dependency databases, use CIUUPDB and for the affinity databases, CIUAFFLD. Before running these jobs, however, review and edit them to make sure they comply with your custom CICS configuration. For detailed editing instructions, see Chapter 6, “Updating the Dependency and Affinity database objects,” on page 121.

Once the database is updated, you can view and analyze the data collection results with the CICS IA Explorer, a graphical user interface to CICS IA.

Viewing Data with CICS IA Explorer

The CICS Explorer with the CICS Interdependency Analyzer plug-in provides an Eclipse-based interface that allows you to analyze your CICS IA data.

For more information about CICS Explorer and the CICS Interdependency Analyzer plug-in, see <https://www.ibm.com/cics/explorer>.

This site will guide you on how to download the CICS Explorer and the CICS IA plugin for the Explorer. As part of the download you will receive the *cicsia_plugin_release_notes.html* and the *cicts_explorer_release_notes.html* files. These files will guide you on how to install and configure the CICS IA Explorer plug-in.

Note: The CICS IA Explorer plugin connects to the CICS IA DB2 database. In order to use the CICS IA plugin, you only need to setup this connection. You do not need to connect the base CICS Explorer to a CICS CPSM environment or a CICS single region using the CMCI connection.

You can find more information about the CICS Explorer plug-in usage in the *IBM CICS Explorer User Guide*.

Chapter 3. Preparing to use CICS IA

This section describes what you need to do before you can use CICS IA.

It contains:

- “Data sets used during this configuration”
- “Running the installation customization program” on page 44
- “Creating VSAM files” on page 48
- “Building the CICS IA database” on page 50
- “Creating your own program exclude, transaction exclude, and resource prefix lists” on page 55
- “Migrating from CICS IA Version 3.1” on page 60
- “Defining resources to CICS” on page 61
- “Tailoring your CICS startup job” on page 65
- “Restarting your CICS regions” on page 66
- “CICS IA supplied modules required in the MVS linklist” on page 66

Data sets used during this configuration

The installation process for CICS IA is described in the *Program Directory* that is distributed with the product. The size of the tar file that is used to install CICS IA is 185 MB.

Installation creates a number of libraries, listed below, that you use to set up and run CICS IA. In the following list, h1q (high-level qualifier) is the library prefix defined in your installation job stream.

CICS IA provides an ISPF-based customization program that you can use to provide values for the variables used by the installation jobs and to create job header cards to meet your site standards. See “Running the installation customization program” on page 44. You are recommended to use this program. The instructions in the topics that follow assume that the customization program has been run. However, sometimes you must edit some of the installation jobs after they have been customized by the installation customization program.

h1q.SCIUSMP1

Contains sample jobs, listed in Table 2, to run the Load Module Scanner and the Reporter and the jobs and files used to set up CICS IA, except for DB2 jobs.

Table 2. Jobs and files supplied in h1q.SCIUSMP1

Member	Description
CIUAFFBL	Job to run the Builder to create combined affinity-transaction-group definitions suitable for input to CICSplex SM.
CIUAFFLD	Job to load the affinities tables, CIU_AFF_CMD_DATA and CIU_AFF_GRP_DATA, with data from the CICS IA affinity files.
CIUAFFLX	Job to load the affinities tables with data in VSAM files gathered by the CICS Transaction Affinities utility.
CIUAFFRD	Job to run the Affinities Reporter against the Affinities database, to report transaction affinities.
CIUAFFRP	Job to run the Affinities Reporter against the CICS IA VSAM affinity data files, to report transaction affinities.

Table 2. Jobs and files supplied in hlq.SCIUSMP1 (continued)

Member	Description
CIUAFFRX	Job to run the CICS Transaction Affinities Reporter against the VSAM files gathered by the CICS Transaction Affinities utility, to report transaction affinities.
CIUDEFF	Definition of local file resources that are used by the Collector.
CIUDEFR	Definition of remote file resources that are used by the Collector.
CIUDEFT	Definition of CSD resources for the Collector.
CIUDEFW	Definition of Web Support resources.
CIUDELGR	Job to delete and remove old groups that contain resource definitions from CICS list.
CIUINDEX	List of all sample JCLs that are contained in SCIUSAMP.
CIUJCDLS	Define the Command Flow log stream data set.
CIUJCFIL	Job to define local files to all CICS TS releases.
CIUJCFIR	Job to define remote files to all CICS TS releases.
CIUJCINT	Job to define CICS resources for the Collector to all CICS TS releases.
CIUJCLCA	Job to create the VSAM dependency files.
CIUJCLCC	Job to create the VSAM control file.
CIUJCLCG	Define the Command Flow GDG data set.
CIUJCLCS	Sample JCL to run the CSECT Scanner.
CIUJCLDL	Sample JCL to run the Cleaner.
CIUJCLLD	Job to run the Load Module Scanner (Detail report).
CIUJCLLS	Job to run the Load Module Scanner (Summary report).
CIUJCLPL	Sample JCL to assemble and link the sample resource prefix list.
CIUJCLRP	Job to run the Dependency Reporter to report dependencies on CICS, DB2, MQ, or IMS resources.
CIUJCLTD	Job to run the Load Module Scanner (Detail report) and to load the DB2 table CIU_SCAN_DETAIL.
CIUJCLTR	Sample JCL to assemble and link the sample TRUE include list.
CIUJCLTS	Job to run the Load Module Scanner (Summary report) and to load the DB2 table CIU_SCAN_SUMMARY.
CIUJCLXP	Sample JCL to assemble and link the sample program exclude list.
CIUJCLXT	Sample JCL to assemble and link the sample transaction exclude list.
CIUJCNNL	Sample JCL to assemble and link-edit the Natural Name List.
CIUJCUPD	Job to run program CIUCFUPD.
CIUJLCPY	Copy Command Flow records from the log stream data set to a sequential GDG data set.
CIUJLDEL	Delete processed Command Flow records from the log stream data set.
CIUJSAMP	Sample JCL to run sample queries.
CIUJTSQ2	Job to run the Threadsafe Analysis report.
CIUJCWEB2	Sample JCL to install Web Support resources.

Table 2. Jobs and files supplied in hlq.SCIUSMP1 (continued)

Member	Description
CIUMIGCD	Job to migrate the VSAM collection datasets.
CIUMIGVD	Job to migrate the VSAM control dataset CIUCNTL.
CIUPRUNE	Job to prune old rows from DB2 tables.
CIURESLD	Job to update the CIU_RESOURCE table.
CIUADB	Job to prepare QSAM CSV files for all basic tables.
CIUADBFAF	Job to create CSV file for the CIU_AFF_EVENTS table.
CIUADBAP	Job to create CSV file from the Applications XML file.
CIUADB4	Create data set of Command Flow records in CSV format.
CIUUPDB	Job to update all types of dependencies, CICS, DB2, MQ, Natural and IMS, in the Dependency database objects from the dependency data files.
CIUUPDBN	Job to update Natural dependencies in the Dependency database objects from the Natural dependency data file.
CIUUPDB1	Job to update CICS dependencies in the Dependency database objects from the CICS dependency data files.
CIUUPDB2	Job to update DB2 dependencies in the Dependency database objects from the DB2 dependency data file.
CIUUPDB3	Job to update MQ dependencies in the Dependency database objects from the MQ dependency data file.
CIUUPDB4	Job to update IMS dependencies in the Dependency database objects from the IMS dependency data file.
CIUUPDB5	Update DB2 table CIU_CMDFLOW_DATA with Command Flow data.
CIUDEFL	Defines CSD resources for the Collector: specifically, group name for CICS resource LIBRARY.
CIUCMDUE	Sample job for compiling and link editing CICS IA Command Flow user exit program and a sample user exit program.

hlq.SCIUSMP2.

Contains sample jobs, listed in Table 2, to set up the data base for CICS IA.

Table 3. Jobs and files supplied in hlq.SCIUSMP2

Member	Description
CIUALOAD	Job to load the CIU_APPLS_DESC and CIU_APPLS_RESOURCES tables.
CIUAPMIG	Job to migrate applications from CICS IA V3.1 to CICS IA V3.2.
CIUCLR	Job to clear the CIU_CICS_DATA, CIU_DB2_DATA, CIU_MQ_DATA, CIU_NATURAL_DATA, and CIU_IMS_DATA tables.
CIUDBCQ	Job to create DB2 tables.
CIUDBCT	Job to create indexes for SYSIBM.
CIUDBLOD	Sample job to run the DB2 utility LOAD against CICS IA data.
CIUDBNB	Job to bind CIUCINB2.
CIUDBNT	Job to bind programs that update DB2 tables.
CIUDBN2	Job to bind program CIUAPEXT to CICS IA V2 tables.

Table 3. Jobs and files supplied in hlq.SCIUSMP2 (continued)

Member	Description
CIUDBORG	Sample job to run the DB2 utility REORG against CICS IA data.
CIUDBSTA	Sample job to run the DB2 utility RUNSTATS against CICS IA data.
CIUDBUNL	Sample job to run the DB2 utility UNLOAD against CICS IA data.
CIUIVPLD	Job to load the Dependency database objects with IVP-supplied data.
CIUMIG31	Sample JCL to migrate DB2 tables from CICS IA V3.1 to CICS IA V3.2.
CIUTLOAD	Job to load the CIU_TRANSLATORS table.
CIUTRLOD	Job to load a TRUE list into the CIU_TRUEEXIT_INFO table
CIUTSLOD	Job to load data into the CIU_THREADSAFE_CMD table.
CIUVERLD	Job to update the CIU_VERSION table.

hlq.SCIUSQL

Contains data files, shown in Table 4, that contain SQL statements used in creating the Dependency and Affinity database objects.

Table 4. Files supplied in hlq.SCIUSQL

Member	Description
CIUAFF	SQL commands to create the CIU_AFF_CMD_DATA and CIU_AFF_GRP_DATA tables and indexes.
CIUAFFV	SQL commands to create the V_CIU_AFFINITY view.
CIUAFFV8	SQL commands to create the CIU_AFF_CMD_DATA and CIU_AFF_GRP_DATA tables and indexes in DB2 V8.
CIUAPPLD	SQL commands to create the CIU_APPLS_DESC and CIU_APPLS_RESOURCES tables and indexes.
CIUCICS1	SQL commands to create the CIU_CICS_DATA table for DB2 Version 7, or DB2 Version 8 in compatibility mode.
CIUCICS7	SQL commands to create the V_CIU_CICS_INDS view.
CIUCICS8	SQL commands to create the CIU_TRUEEXIT_INFO table and indexes.
CIUCICS9	SQL commands to create the V_CIU_TRUEEXIT_INFO view.
CIUCICSX	SQL commands to create the CIU_CICS_DATA table for DB2 Version 8 and later. Do not use this command with DB2 Version 8 running in compatibility mode.
CIUCSSD	SQL commands to create the CIU_PROGRAM_INFO, CIU_CSECT_INFO, and CIU_TRANSLATORS tables and indexes.
CIUCSSV	SQL commands to create the V_CIU_CICS_LINKED and V_CIU_CSECT_TRANS tables.
CIUDB2	SQL commands to create the CIU_DB2_DATA table and indexes.
CIUDB2V	SQL commands to create the V_CIU_DB2_RES and V_CIU_DB2_RES2 views.

Table 4. Files supplied in hlq.SCIUSQL (continued)

Member	Description
CIUGRNTC	SQL commands to grant access to the CICS plan.
CIUGRNTB	SQL commands to grant access to the batch plan.
CIUIBM1	SQL commands to build indexes for the SYSIBM tables.
CIUIMS	SQL commands to create the CIU_IMS_DATA table and indexes.
CIULMSD	SQL commands to create the CIU_SCAN_SUMMARY and CIU_SCAN_DETAIL tables and indexes.
CIULOADC	DB2 LOAD utility statements to load CICS IA V3.2 CIU_CICS_DATA table.
CIULOADD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_DB2_DATA table.
CIULOADI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_IMS_DATA table.
CIULOADM	DB2 LOAD utility statements to load CICS IA V3.2 CIU_MQ_DATA table.
CIULODAC	DB2 LOAD utility statements to load CICS IA V3.2 CIU_AFF_CMD_DATA table.
CIULODAF	DB2 LOAD utility statements to load CICS IA V3.2 CIU_AFF_EVENTS table.
CIULODAG	DB2 LOAD utility statements to load CICS IA V3.2 CIU_AFF_GRP_DATA table.
CIULODAI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_AFF_INDEX table.
CIULODCI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_CSECT_INFO table.
CIULODEI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_EXIT_INFO table.
CIULODFD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_FILE_DETAIL table.
CIULODPD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_PROGRAM_DETAIL table.
CIULODPI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_PROGRAM_INFO table.
CIULODRI	DB2 LOAD utility statements to load CICS IA V3.2 CIU_REGION_INFO table.
CIULODSD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_SCAN_DETAIL table.
CIULODSS	DB2 LOAD utility statements to load CICS IA V3.2 CIU_SCAN_SUMMARY table.
CIULODTD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_TDQUEUE_DETAIL table.
CIULODTR	DB2 LOAD utility statements to load CICS IA V3.2 CIU_TRANSID_DETAIL table.
CIULODTS	DB2 LOAD utility statements to load CICS IA V3.2 CIU_TSQUEUE_DETAIL table.
CIULODWD	DB2 LOAD utility statements to load CICS IA V3.2 CIU_WEBSERV_DETAIL table.

Table 4. Files supplied in hlq.SCIUSQL (continued)

Member	Description
CIULOADN	DB2 LOAD utility statements to load CICS IA V3.2 CIU_NATURAL_DATA table.
CIUMIGAC	SQL statements to migrate CIU_AFF_CMD_DATA table.
CIUMIGAD	SQL statements to migrate CIU_APPLS_DESC table.
CIUMIGAG	SQL statements to migrate CIU_AFF_GRP_DATA table.
CIUMIGAI	SQL statements to migrate CIU_AFF_INDEX table.
CIUMIGAR	SQL statements to migrate CIU_APPLS_RESOURCE table.
CIUMIGCI	SQL statements to migrate CIU_CSECT_INFO table.
CIUMIGEI	SQL statements to migrate CIU_EXIT_INFO table.
CIUMIGFD	SQL statements to migrate CIU_FILE_DETAIL table.
CIUMIGLC	SQL statements to migrate CIU_CICS_DATA table.
CIUMIGLD	SQL statements to migrate CIU_DB2_DATA table.
CIUMIGLI	SQL statements to migrate CIU_IMS_DATA table.
CIUMIGLM	SQL statements to migrate CIU_MQ_DATA table.
CIUMIGLN	SQL statements to migrate CIU_NATURAL_DATA table.
CIUMIGPD	SQL statements to migrate CIU_PROGRAM_DETAIL table.
CIUMIGPI	SQL statements to migrate CIU_PROGRAM_INFO table.
CIUMIGRI	SQL statements to migrate CIU_REGION_INFO table.
CIUMIGSD	SQL statements to migrate CIU_SCAN_DETAIL table.
CIUMIGSS	SQL statements to migrate CIU_SCAN_SUMMARY table.
CIUMIGTD	SQL statements to migrate CIU_TDQUEUE_DETAIL table.
CIUMIGTR	SQL statements to migrate CIU_TRANSID_DETAIL table.
CIUMIGTS	SQL statements to migrate CIU_TSQUEUE_DETAIL table.
CIUMIGWD	SQL statements to migrate CIU_WEBSERV_DETAIL table.
CIUM1331	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V1.3 to CICS TS V3.1.
CIUM2231	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.2 to CICS TS V3.1.
CIUM2331	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.3 to CICS TS V3.1.
CIUM1332	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V1.3 to CICS TS V3.2.
CIUM2232	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.2 to CICS TS V3.2.
CIUM2332	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.3 to CICS TS V3.2.

Table 4. Files supplied in hlq.SCIUSQL (continued)

Member	Description
CIUM3132	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V3.1 to CICS TS V3.2.
CIUM1341	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V1.3 to CICS TS V4.1 or to CICS TS V4.2.
CIUM2241	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.2 to CICS TS V4.1 or to CICS TS V4.2.
CIUM2341	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V2.3 to CICS TS V4.1 or to CICS TS V4.2.
CIUM3141	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V3.1 to CICS TS V4.1 or to CICS TS V4.2.
CIUM3241	Queries to identify programs that include deleted or renamed API commands, SPI commands, or both when migrating from CICS TS V3.2 to CICS TS V4.1 or to CICS TS V4.2.
CIUMAIN	SQL commands to delete and create the Dependency database objects.
CIUMQ1	SQL commands to create the CIU_MQ_DATA table and indexes.
CIUNAT	SQL commands to create the CIU_NATURAL_DATA table and a unique index.
CIUPCICS	SQL commands to prune data from the CIU_CICS_DATA table.
CIUPDB2	SQL commands to prune data from the CIU_DB2_DATA table.
CIUPEXIT	SQL commands to prune data from the CIU_EXIT_INFO table.
CIUPFILE	SQL commands to prune data from the CIU_FILE_DETAIL table.
CIUPIMS	SQL commands to prune data from the CIU_IMS_DATA table.
CIUPMQ	SQL commands to prune data from the CIU_MQ_DATA table.
CIUPPROG	SQL commands to prune data from the CIU_PROGRAM_DETAIL table.
CIUPTDQ	SQL commands to prune data from the CIU_TDQUEUE_DETAIL table.
CIUPTRAN	SQL commands to prune data from the CIU_TRANSID_DETAIL table.
CIUPTSQ	SQL commands to prune data from the CIU_TSQUEUE_DETAIL table.
CIUPWEBS	SQL commands to prune data from the CIU_WEBSERV_DETAIL table.
CIURCMD	SQL commands to create the Command Flow CIU_CMDFLOW_DATA table.
CIUREGTB	SQL commands to create the CIU_REGION_INFO table.
CIURESIN	SQL commands to reload the CIU_RESOURCE table.
CIURESTB	SQL commands to create the CIU_RESOURCE table for DB2 Version 7.

Table 4. Files supplied in hlq.SCIUSQL (continued)

Member	Description
CIURESTX	SQL commands to create the CIU_RESOURCE table for DB2 Version 8 and later.
CIUREXIT	SQL commands to create CIU_EXIT_INFO table.
CIURFILE	SQL commands to create CIU_FILE_DETAIL table.
CIURPROG	SQL commands to create CIU_PROGRAM_DETAIL table.
CIURTDQ	SQL commands to create CIU_TDQUEUE_DETAIL table.
CIURTRAN	SQL commands to create CIU_TRANSID_DETAIL table.
CIURTSQ	SQL commands to create CIU_TSQUEUE_DETAIL table.
CIURWEB	SQL commands to create CIU_WEBSERV_DATA table.
CIUSAMP0 - CIUSAMP4, CIUSAMP6 - CIUSAMP9, CIUSAMPA, CIUSAMPB, CIUSAMPC, CIUSAMPD, CIUSAMPE, CIUSAMPN, CIUSPACE, CIUSPCAF, CIUSPCCS, CIUSPCDB, CIUSPCDN, CIUSPCD1 - CIUSPCD4, CIUSPCTD, CIUSPCTS	Sample SQL queries.
CIUSDAPP	SQL commands to create the CIUSPAPP stored procedure.
CIUSTSV	SQL commands to create the V_CIU_SCAN_TRDSAFE table.
CIUTSCTB	SQL commands to create the CIU_THREADSafe_CMD table.
CIUVER	SQL commands to create the CIU_VERSION table.
CIUVERUP	SQL commands to update the CIU_VERSION table.
CIURESTL	SQL commands to load the CIU_RESOURCE table.
CIURESTU	SQL commands to unload Dependency resource data.
CIUREVNT	SQL to generate the CIU_EVENT_DETAIL table and associated view.

Running the installation customization program

The installation customization program helps you to customize the CICS IA sample jobs, clists, and SQL definitions. It creates customized installation jobs in which the names of system entities, such as the high-level qualifier (hlq) of the CICS IA data sets, are set to specified values to suit your local environment.

Appendix E, "Worksheet for the installation customization program," on page 317 contains a worksheet for use with the installation customization program. The worksheet consists of a table of installation variables, such as `_dbid_`, the identifier of the DB2 database, that can be passed to the installation customization program. You can record the value that you assign to each variable in the "Value" column of the table.

Before running the installation customization program talk to the relevant systems personnel to gather the information about the local system environment, such as the high-level qualifiers of data sets that you need to run the installation customization program. Use the worksheet as a memory jogger, and to record the values that you need.

To invoke the installation customization program, run member CIUCNFG1 of the SCIUEXEC library. The program requires two parameters to be passed to it:

1. The high-level qualifier of the CICS IA data sets &HLQ.
2. The national language to be used in messages and on panels &LANG. Two languages are supported:

ENU American English (the default)

JPN Japanese

For example, you could invoke the installation customization program by entering the following at an ISPF command line:

```
tso ex 'CICSIA.V320.SCIUEXEC(CIUCNFG1)' 'CICSIA.V320 ENU'
```

Customized members of each CICS IA library are saved as follows:

- Customized members of '&HLQ.SCIUCLIS' in '&HLQ.SCIUCLIS.OUT'
- Customized members of '&HLQ.SCIUSAME', and '&HLQ.SCIUSAMK' in '&HLQ.SCIUSMP1.OUT', '&HLQ.SCIUSMP2.OUT', '&HLQ.SCIUSAME.OUT' and '&HLQ.SCIUSAMK.OUT'
- Customized members of '&HLQ.SCIUSQL' in '&HLQ.SCIUSQL.OUT'

The members of '&HLQ.SCIUDAT1' and '&HLQ.SCIUDAT2' are not customized but are copied to '&HLQ.SCIUDAT1.OUT' and '&HLQ.SCIUDAT2.OUT'.

Note:

1. For convenience, in this and the following sections the customized output files are given the names of their respective input files with a suffix of "OUT". However, you can change the names of the output files allowing you to create configured data for different CICSplexes and different levels of CICS.
2. You cannot change the names of some system entities using the installation customization program. For lists of those that can and those that cannot, see Appendix E, "Worksheet for the installation customization program," on page 317.

Starting CICS IA customization

You can customize CICS IA by creating new configurations or changing parameters of the configurations that already exist.

CICS IA stores configuration information in two ISPF tables. These tables are CIUCICST and CIUDB2T. The first time you use the configuration utility, it creates these tables in a dataset that is called *userid*.ISPTABL. If you are migrating from CICS IA V3.1, it will take your existing configurations, which were stored in a dataset called *userid*.ISPTABL(*userid*), and create the two new tables in the same dataset. You can use the SETTINGS option to rename the dataset that holds the ISPF tables. This enables you to share your configurations with other users.

To customize CICS IA, on the Welcome panel, CIUCNF00, press Enter to proceed to the CIUCNF04 panel, shown in Figure 6 on page 46.

```
CIUCNF04 ***** CICS Interdependency Analyzer for z/OS - V3R2M0 *****
```

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

```
Please select how would you like to configure the region:
```

- 0. SETTINGS **0**
- 1. Create new configuration **1**
- 2. Use configuration from existing list **2**

```
Command ==>
```

Figure 6. Panel CIUCNF04

Notes:

0 Settings

To define the library for storing the ISPF table, type action code 0 and press Enter. Panel CIUCNF07, shown in Figure 9 on page 48, is displayed.

1 Create new configuration

To create a new configuration, type action code 1 on panel CIUCNF04 and press Enter. Panel CIUCNF06, shown in Figure 8 on page 47, is displayed.

2 Use configuration from existing list

To choose one of the existing configurations or change its parameters, type action code 2 and press Enter. Panel CIUCNF05, shown in Figure 7 on page 47, is displayed.

Using the configurations from the existing list

If you want to use one of the existing configurations, select this configuration from the list displayed on panel CIUCNF05, shown in Figure 7 on page 47.

CIUCNF05 ***** CICS Interdependency Analyzer for z/OS - V3R2M0 Row 1 to 3 of 3

CICS IA Customization Function

Available configurations:

	Cmd	Configuration	DB2Version	Description
1		c_w4	V810	my config
		V1	v810	v2
		IYDZZ32F	V810	my config

***** Bottom of data *****

Command ==>

Scroll ==> PAGE

Figure 7. Panel CIUCNF05

Note:

1 Cmd

You can use this field for the following actions:

- To select one of the configurations represented, type action code s next to the configuration name and press Enter.
- To delete a configuration, type d and press Enter.
- To copy an existing configuration and all its parameters to a new configuration, type action code c and press Enter. The new configuration is displayed in the configurations list and the first two symbols of its name are replaced with “c_”.

Creating a new configuration

When creating a new configuration, fill in the appropriate fields on panel CIUCNF06, shown in Figure 8.

CIUCNF06 ***** CICS Interdependency Analyzer for z/OS - V3R2M0 *****

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

Please enter configuration name, DB2 version and description:

Configuration . . . **1**
DB2version . . . **2**
Description . . . **3**

Command ==>

Figure 8. Panel CIUCNF06

Notes:

1 Configuration

Type the name of the configuration that you intend to create.

2 DB2version

Choose the version of DB2 for the new configuration.

3 Description

Enter a short description of the new configuration.

Defining the ISPF table storing library

When defining a library to store the ISPF table, fill in the appropriate fields on panel CIUCNF07, shown in Figure 9.

```
CIUCNF07 ***** CICS Interdependency Analyzer for z/OS - V3R2M0 *****
Command = = = > _____

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

Please specify settings:

CICS IA ISPTABL . . . . . 1

. . . . .
```

Figure 9. Panel CIUCNF07

Note:

1 CICS IA ISPTABL

Enter the name of the library in which you want the ISPF table to be stored. This library can be used to share configurations with other users.

Creating VSAM files

The VSAM files that CICS IA uses are listed below.

For VSAM file space allocation refer to “VSAM data set allocation” on page 331.

Table 5. 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. An unrecoverable file used to hold control information.	CIUJCLCC

Table 5. CICS IA VSAM files and associated jobs (continued)

File	Description	Job
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
hlq.CIUINT7	The Natural dependency data file. An unrecoverable file used to record dependencies on Natural resources.	CIUJCLCA

To control the operation of multiple instances of the Collector, running on different CICS regions, from a single CICS terminal, you must share the dependency and affinity data files and the control record file across all the regions being monitored. To share the files use either:

- VSAM record-level sharing (RLS). For information about using VSAM RLS in CICS, see the *CICS Installation Guide*.
- Function shipping to a file-owning region (FOR). For information about CICS function shipping, see the *CICS Intercommunication Guide*. If you use function shipping, it is recommended that, for performance reasons, you define the files as local to the region in which you expect to do most monitoring activity and remote in all the other regions to be monitored.

To migrate the VSAM control record file that you used in CICS IA Version 1.3, Version 2.1, Version 2.2 or Version 3.1 to CICS IA Version 3.2, use one of the following methods:

- Run the job that is described in “Migrating the VSAM control record file” on page 61. You must first follow the steps in this section to create empty dependency and affinity data files and a control record file in CICS IA Version 3.2 format.
- Run the CIUJCLCC job in hlq.SCIUSMP1.OUT.

For VSAM file space allocation refer to “VSAM data set allocation” on page 331.

To create the dependency and affinity data files, and the control record file, review and run the following jobs:

- hlq.SCIUSMP1.OUT(CIUJCLCA)
- hlq.SCIUSMP1.OUT(CIUJCLCC)

Creating Log Streams and GDGs

This section describes the process of collecting and updating command flow data.

To get the latest command flow data, you should complete two basic tasks:

1. Review and run the sample CIUJCDLS job that collects the data, and
2. Review and run the sample CIUJCLCG job that defines the attributes for the command flow GDG dataset.

Both these tasks are covered in the sections that follow.

Before you start collecting data, consider the possible storage options for the CICS IA log stream:

1. Coupling facility, where log stream data is duplexed to a logger data space or staging data set. The coupling facility allows data from multiple CICS regions running on multiple z/OS images to log command flow data to the same log stream
2. DASD-only, where log stream data is duplexed in the MVS logger data space. The DASD-only option allows data from multiple CICS regions running on the same z/OS image to log command flow data to the same log stream

Editing and running the CIUJCDLS job

The sample CIUJCDLS job can be found in hlq.SCIUSMP1.OUT.

Before running the job, make sure that

- the value defined for the logstream name matches the one specified in the CICS JOURNALMODEL definition in hlq.SCIUSMP1(CIUDEFT).

Note: The logstream name and its attributes may be changed to meet your system requirements.

- the value for MAXBUFSIZE is not larger than 32760.

Editing and running the CIUJCLCG job

The sample CIUJCLCG job can be found in hlq.SCIUSMP1.OUT.

Command flow records are recorded to a MVS log stream data set. Once collected they are copied to a GDG dataset that you can then use to update the CIU_CMDFLOW_DATA table or CSV file.

To define the attributes for the command flow GDG dataset, run the sample CIUJCLCG job.

Note: You can set the LIMIT option to your system requirements.

Building the CICS IA database

This section describes how to set up the Dependency, Affinity, and Load Module Scanner databases objects.

There is one set of Dependency database objects, even if you have separate dependency data and control record files for each region monitored by the Collector. Typically, you feed the dependency information in the dependency data

files for each region into the Dependency database objects, which helps you to compare and contrast dependency data from different regions.

Similarly, there is one set of Affinity database objects, even if you have separate affinity data files for each region monitored by the CICS IA Collector. Typically, you feed the affinity information in the affinity data files for each region into the Affinity database objects, which helps you to compare and contrast affinity data from different regions.

Because there is typically only one set of Dependency and one set of Affinity database objects, which contain the dependency and affinity data for all the monitored regions, you can use one instance of the Query interface to access, compare, and contrast dependency and affinity data from different regions.

Defining the database

There are the JCL that you must edit to define the database that conforms with your own system conventions.

About this task

Before you run the installation customization program, read “DB2 considerations.”

For DB2 tablespace and index allocations, refer to “DB2 space allocation” on page 333.

To define the Dependency, Affinity, and Scanner databases:

1. Review the following members in hlq.SCIUSMP2.OUT:
 - CIUDBCQ
 - CIUDBNB
 - CIUVERLD
2. Review the associated SQL members for the above jobs. They are in hlq.SCIUSQL.OUT.
3. If necessary, run hlq.SCIUSMP2.OUT(CIUDBCQ) to create the database tables.
4. Run hlq.SCIUSMP2.OUT(CIUDBNB) to bind CIUCINB2.

DB2 considerations

Before running the installation jobs to create the DB2 environment, decide the following:

- Under which qualifier to create DB2. This qualifier is used to tie the database tables to the applications at bind time. The customization variable *_qual_* is set to this value throughout all the installation jobs.
- The owner of the DB2 plans. The customization variable *_own_* is set to this value throughout all the installation jobs.
- How to grant access to the plans and tables. See “Granting access to the plans and tables” on page 52.
- For other DB2 variables, refer to Appendix E, “Worksheet for the installation customization program,” on page 317.

DB2 versions

CICS IA supports DB2 Version 7 and above. If you are planning to install CICS IA with DB2 Version 8 in compatibility mode when you use the supplied configuration exec select V710 as your DB2 version.

DB2 access

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 SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT DB2 tables. You can get this access by giving the userid access to the CICS IA plan for the CINB transaction. See section “Granting access to the plans and tables.”

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 it might be the PLT user ID 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.

The batch jobs to load all CICS IA database use the DB2 LOAD and UNLOAD utilities. The RACF userid or group used for these jobs will require DB2 ADM authority on the CICS IA database.

Read this section in conjunction with the *CICS DB2 Guide*

Granting access to the plans and tables:

The CICS system programmer and the DB2 administrator must decide how to control access to the CICS IA plan and the CIU tables.

About this task

CICS IA uses both static and dynamic SQL, you need more than just a GRANT EXECUTE ON PLAN CICSIA TO PUBLIC command. You must also allow dynamic SQL requests to access the tables. You have two options:

Option 1

Specify the DYNAMICRULES(BIND) option on the BIND PLAN command in CIUDBNB and CIUDBNT. This option is recommended for these reasons:

- The way in which security works is the same for both dynamic and static SQL.
- If you grant permissions by issuing one or more GRANT EXECUTE ON PLAN CICSIA TO _xxxx_ commands, all security checks are done at the plan level; this option is simple to administer and offers good performance.
- If, as is typically the case, the _xxxx_ in the GRANT EXECUTE command specifies a RACF group rather than a single RACF user ID, to add new users you just connect the users to the RACF group.

The sample installation jobs CIUDBNB and CIUDBNT use the DYNAMICRULES(BIND) option and then issue GRANT EXECUTE commands for the appropriate plans. These commands are issued against a RACF group.

To use this option:

1. Select your RACF group.
2. Change _racfgrp_ to your chosen RACF group.
3. Ensure that all CIU users are connected to your chosen RACF group, with RACF “list of groups” active in the system.
4. Enable secondary authorization in DB2. See the DB2 install job DSNTIJEX.

Note:

1. Review DSNTIJEX job with your DB2 administrator.

2. For a full understanding of the implications of DYNAMICRULES(BIND), see the description of the BIND COMMAND in the *DB2 Commands* manual.
3. See also the section on DB2 security in the *CICS RACF Security Guide*.
4. Review this job with your DB2 administrator.

Option 2

Grant all CIU users access to the tables explicitly. This option is not recommended because you have to do this every time you give access to a new user.

To use this option:

1. In the sample jobs CIUDBNB and CIUDBNT, on the BIND PLAN command change the DYNAMICRULES option from DYNAMICRULES(BIND) to DYNAMICRULES(RUN).
2. In hlq.SCIUSQL.OUT(CIUGRNTC), change the sample GRANT commands to GRANT EXECUTE on the CICSIA plan and GRANT SELECT, GRANT UPDATE, GRANT INSERT, GRANT DELETE, and any other GRANT commands, on the CIU tables.
3. If the GRANT permissions are made to a RACF group, note these requirements:
 - a. Ensure that all CIU users are connected to that RACF group.
 - b. Enable secondary authorization in DB2. See the DB2 install job DSNTIJEX for more information if required.

The Query interface uses dynamic SQL to access the CIU_CICS_DATA, CIU_DB2_DATA, CIU_MQ_DATA, and CIU_IMS_DATA tables. For guidance on using dynamic SQL with CICS, refer to the *CICS DB2 Guide* for your CICS release.

The delivered SQL is constructed and sized for a default application. You must tailor the sizings for PRIQTY and SECQTY in the index creation batch job to suit your requirements. If you create a new query you must carry out an evaluation to ensure that the existing indexing supports the query. If the existing indexing does not support the query, you must construct additional indexes. Contact your Database administrator if you require assistance.

DB2 Stored Procedures setup

CICS IA V3.2 uses DB2 Stored Procedures to perform complex DB2 tasks.

A stored procedure is an executable code that can be called by other programs. You might choose to use stored procedures for the code that is used repeatedly. Other benefits of stored procedures include network traffic reduction, result sets returned directly to an application, or access to data without granting the privileges to the applications.

DB2 Stored Procedures run in a started task called *WLM* (Work Load Manager) associated with each DB2 subsystem. Sample JCL for a stored procedure can be found in SCIUSMP1(CIUSPTSK). The name of the started task must match the one supplied in the CICS IA configuration variable *_wlm_*.

You can find more information about implementing DB2 Stored Procedures in the *DB2 Version x.1 for z/OS Administration Guide*.

The DB2 Stored Procedures used by CICS IA are of the type *External SQL Stored Procedures*. For details about available stored procedures, see CICS IA External Interfaces.

Grouping transactions and programs into applications

CICS IA allows you to group transactions, programs, or both to form a notional "application".

To form a notional "application", define an 8-character application code, an application description, and the transactions and programs that make up the application. You can then use the CICS IA plug-in to query an application.

Defining new applications

You can define an application with XML verbs.

About this task

All applications are stored in the CIUAPPLS member in your customized SCIUDAT1 data set. They are defined in a simple XML format. After the installation and customization CIUAPPLS contains the CICS IA IVP application. Figure 10 shows the IVP application.

```
<form>
<appl>
<code>IVP      </code>
<name>IVP Application      </name>
<tran>TSTS</tran>
<tran>TWEB</tran>
<prog>EMSTESTS</prog>
<prog>EMSWEBTS</prog>
</appl>
</form>
```

1
2
3
4
5
6

Figure 10. IVP application

Notes:

The XML that is used to describe an application contains the following verbs:

- 1** `<form>` and `</form>` are the start and end verbs for all of the applications.
- 2** `<appl>` and `</appl>` are the start and end verbs for each application.
- 3** `<code>` and `</code>` define the 8-character application code.
- 4** `<name>` and `</name>` define the 50-character application name.
- 5** `<tran>` and `</tran>` define a transaction to the application.
- 6** `<prog>` and `</prog>` define a program to the application.

Running CIUALOAD to define the IVP and your applications

After you have defined all your applications in SCIUDAT1(CIUAPPLS), you can load them into your DB2 tables. To perform this task, review and run sample job CIUALOAD.

Loading static DB2 tables

CICS IA uses static tables to hold translator and threadsafe information.

About this task

To load the threadsafe table, renew and run job SCIUSMP2.OUT(CIUTSLOD).

To load the translator table, review and run job SCIUSMP2.OUT(CIUTLOAD).

Creating your own program exclude, transaction exclude, and resource prefix lists

You can use program exclude lists, transaction exclude lists, and resource prefix list to limit the volume of information collected by the Collector. In this section the term, Collector, refers to the Affinity and Dependency collector functions. The User Command Flow collector function does not reference the exclude lists.

A program exclude list contains a list of program-name prefixes; the Collector does not collect data for any program that has the name beginning with one of the prefixes. Similarly, a transaction exclude list contains a list of transaction-name prefixes; the Collector does not collect data for any transaction that has a name beginning with one of the prefixes.

For details of how to specify which exclude lists the Collector is to use, see “Specifying region-specific options: general” on page 93.

CICS IA supplies sample program exclude, transaction exclude, and resource prefix lists:

- The default program exclude list, supplied with CICS IA, is called CIUXPROG; it contains the name prefixes of IBM components about which you do not normally want to collect information.
- The default transaction exclude list, supplied with CICS IA, is called CIUXTRAN and is empty.
- The default program prefix list, supplied with CICS IA, is called CIUPFXTB and is empty.

Creating a program exclude list

A program exclude list is a load module that contains a simple list of program name prefixes.

About this task

Each list item consists of a 1-byte length field, followed by the characters of the program name prefix. The length is the number of characters in the prefix, which must be in the range 1 through 8. A length of zero indicates the end of the list. Figure 11 on page 56 is an example of a program exclude list.

```

MYXPROG CSECT
MYXPROG AMODE 31
MYXPROG RMODE ANY
        DS    0F
        DC    AL1(8),C'TEST'      Excludes a program called TEST
        DC    AL1(4),C'TEST'      Excludes names starting with TEST
        DC    AL1(3),C'UCC'        Excludes names starting with UCC
        DC    AL1(0)                End of list
END    MYXPROG

```

Figure 11. Example program exclude list

Running the sample batch job CIUJCLXP

Procedure

1. A sample batch job, CIUJCLXP, is provided to assemble and link-edit the sample program exclude list, CIUXPROG. Before running the CIUJCLXP job, change the following:

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The PGM keyword of the EXEC statement of the ASM step

Insert the name of the assembler to use.

The SYSIN DD statement

Specify the name of the assembler language source library where your exclude list is to be found. The default is hlq.SCIUSRCE, where “hlq” is the data set qualifier assigned during installation.

Change the member name to the name of your own program exclude list.

The SYSLMOD DD statement

Specify the name of the CICS IA load library where the exclude list is to be placed. The default is hlq.SCIULOAD, where “hlq” is the data set qualifier assigned during installation.

Change the member name to the name of your own program exclude list.

2. To make the exclude list from your customized program available to the Collector:
 - a. Place the generated load module in a load library concatenated with DDNAME DFHRPL.
 - b. Define the generated load module to CICS, using the same attributes as those used for CIUXPROG in the CIUJnnCR sample JCL in the CICS IA load library. In particular, specify RELOAD(NO) on the PROGRAM definition.

Creating a transaction exclude list

A transaction exclude list is a load module that contains a simple list of transaction name prefixes.

About this task

Each list item consists of a 1-byte length field, followed by the characters of the transaction name prefix. The length is the number of characters in the prefix, which must be in the range 1 through 4. A length of zero indicates the end of the list. Figure 12 on page 57 is an example of a transaction exclude list.

```

MYXTRAN CSECT
MYXTRAN AMODE 31
MYXTRAN RMODE ANY
        DS    0F
        DC    AL1(1),C'C'           Excludes names starting with C
        DC    AL1(3),C'UCC'        Excludes names starting with UCC
        DC    AL1(0)                End of list
        END  MYXTRAN

```

Figure 12. Example transaction exclude list

A sample batch job, CIUJCLXT, is provided to assemble and link-edit the sample transaction exclude list, CIUXTRAN.

Before running the CIUJCLXT job, change the following:

1.

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The PGM keyword of the EXEC statement of the ASM step

Insert the name of the assembler to use.

The SYSIN DD statement

Specify the name of the assembler language source library where your exclude list is to be found. The default is h1q.SCIUSRCE, where “h1q” is the data set qualifier assigned during installation.

Change the member name to the name of your own transaction exclude list.

The SYSLMOD DD statement

Specify the name of the CICS IA load library where the exclude list is to be placed. The default is h1q.SCIULOAD, where “h1q” is the data set qualifier assigned during installation.

Change the member name to the name of your own transaction exclude list.

2.

To make your customized transaction exclude list available to the Collector:

- a. Place the generated load module in a load library concatenated with DDNAME DFHRPL.
- b. Define the generated load module to CICS, using the same attributes as those used for CIUXTRAN in the CIUJnnCR sample JCL in the CICS IA load library. In particular, specify RELOAD(NO) on the PROGRAM definition).

Creating a resource prefix list

A resource prefix list is a load module that contains a simple list of TSQueue and ENQ/DEQ resource name prefixes that helps to avoid collecting unnecessary information.

About this task

Each list item consists of a 1-byte length field, followed by the characters of the resource name prefix. The length is the number of characters in the prefix, which must be in the range 1 - 32. A length of zero indicates the end of the list. Figure 13 on page 58 is an example of a resource name prefix list.

```

CIUPFXTB CSECT
CIUPFXTB AMODE 31
CIUPFXTB RMODE ANY
        DS 0F
Add user prefixes here
        DC AL1(4),C'TEST'           Example
Predefined prefixes
        DC AL1(3),C'DFH' CICS
        DC AL1(0)                   End of list
END CIUPFXTB

```

Figure 13. Example resource name prefix list

Running the sample batch job CIUJCLPL

Procedure

1. A sample batch job, CIUJCLPL, is provided to assemble and link-edit the sample resource prefix list, CIUPFXTB. Before running the CIUJCLPL job, change the following:

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The PGM keyword of the EXEC statement of the ASM step

Insert the name of the assembler to use.

The SYSIN DD statement

Specify the name of the assembler language source library where your resource prefix list is to be found. The default is hlq.SCIUSRCE, where hlq is the data set qualifier assigned during installation.

Change the member name to the name of your own program resource prefix list.

The SYSLMOD DD statement

Specify the name of the CICS IA load library where the resource prefix list is to be placed. The default is hlq.SCIULOAD, where hlq is the data set qualifier assigned during installation.

Change the member name to the name of your own resource prefix list.

2. To make your customized resource prefix list available to the Collector:
 - a. Place the generated load module in a load library concatenated with DDNAME DFHRPL.
 - b. Define the generated load module to CICS, using the same attributes as those used for CIUPFXTB in the CIUDEFT sample JCL in the CICS IA load library. In particular, specify RELOAD(NO) on the PROGRAM definition.

Creating your own TRUE include list

You can use your own include list to choose which TRUES are to be captured.

An include list contains names of the TRUES and names of the products, which issue these TRUES. The Collector collects data for these pairs of TRUE and corresponding product names.

CICS IA supplies a sample TRUE include list. The name of the default list is CIUXITTB, it contains the pairs of TRUE names and names of the products about which you normally want to collect information.

Creating a TRUE include list

A TRUE include list is a load module that contains a simple list of TRUE names and names of the products, which use these TRUES.

About this task

Each list item consists of a 1-byte length field, followed by the characters of the TRUE name and name of the product, which matches this TRUE. The length consists of the number of characters in the TRUE name, which must be in the range 1 through 8, and the number of characters in the corresponding product information, which is 50 characters or more. A length of zero indicates the end of the list. Figure 14 is an example of a TRUE include list.

```
MYXITTB CSECT
MYXITTB AMODE 31
MYXITTB RMODE ANY
        DS      0F
        DC      AL1(8),C'TEST'      Includes TRUE called TEST
        DC      AL1(4),C'TEST'      Includes names starting with TEST
        DC      AL1(3),C'UCC'       Includes names starting with UCC
        DC      AL1(0)              End of list
        END    MYXITTB
```

Figure 14. Example TRUE include list

Running the sample batch job CIUJCLTR

Procedure

1. A sample batch job, CIUJCLTR, is provided to assemble and link-edit the sample TRUE include list, CIUXITTB. Before running the CIUJCLTR job, change the following:

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The PGM keyword of the EXEC statement of the ASM step

Insert the name of the assembler to use.

The SYSIN DD statement

Specify the name of the assembler language source library where your include list is to be found. The default is h1q.SCIUSRCE, where “h1q” is the data set qualifier assigned during installation.

Change the member name to the name of your own TRUE include list.

The SYSLMOD DD statement

Specify the name of the CICS IA load library where the include list is to be placed. The default is h1q.SCIULOAD, where “h1q” is the data set qualifier assigned during installation.

Change the member name to the name of your own TRUE include list.

2. To make your customized TRUE include list available to the Collector:
 - a. Place the generated load module in a load library concatenated with DDNAME DFHRPL.
 - b. Define the generated load module to CICS, using the same attributes as those used for CIUXITTB in the CIUDEFT sample JCL in the CICS IA load library. In particular, specify RELOAD(NO) on the PROGRAM definition.

Migrating from CICS IA Version 3.1

Four tasks are involved in migrating from previous versions of CICS IA.

This section describes how to migrate from CICS IA Version 3.1 to CICS IA Version 3.2.

1. "Removing old definitions"
2. "Migrating the DB2 tables"
3. "Migrating the VSAM control record file" on page 61
4. "Migrating application definitions" on page 61

Removing old definitions

You must remove any resource definitions for previous CICS IA versions from CICS when you migrate to a later version.

To remove resource definitions from previous CICS IA versions:

1. Edit the CIUDELGR job in hlq.SCIUSMP1.OUT and set the lines as shown in Table 6.

Table 6. Editing the CIUDELGR job to remove old CICS IA definitions from CICS TS

CICS IA version from which you are migrating	Lines to be edited
CICS IA Version 1.1	DELETE GROUP(INxxGRP) REMOVE GROUP(INxxGRP)
CICS IA Version 1.2	DELETE GROUP(CIUxxGRP) REMOVE GROUP(CIUxxGRP)
CICS IA Version 1.3	DELETE GROUP(CIUxxG13) REMOVE GROUP(CIUxxG13)
CICS IA Version 2.1	DELETE GROUP(CIUxxG21) REMOVE GROUP(CIUxxG21)
CICS IA Version 2.2	DELETE GROUP(CIUxxG22) REMOVE GROUP(CIUxxG22)

2. Run hlq.SCIUSMP1.OUT(CIUDELGR).
3. Remove the previous CICS IA Version SCIULOAD library from the CICS startup JCL.

Migrating the DB2 tables

CICS IA Version 3.2 provides sample jobs to migrate your CICS IA V3.1 DB2 tables to CICS IA V3.2 tables.

To use the migrate job you must have created the new CICS IA V3.2 tables as described in "Defining the database" on page 51.

- CICS, DB2, MQ and IMS tables
- Load Module and CSECT Scanner tables
- Affinity tables
- Detailed CICS resource tables

To migrate these tables from CICS IA V3.1 to CICS IA V3.2, review and run the hlq.SCIUSMP2.OUT(CIUMIG31) job.

Migrating the VSAM control record file

To migrate the VSAM control record file that you used in CICS IA Version 3.1 to CICS IA Version 3.2, review and run the hlq.SCIUSMP1.OUT(CIUMIGVD) job.

Migrating the VSAM collection datasets

To migrate the VSAM collection datasets used in CICS IA Version 3.1 to CICS IA Version 3.2, review and run the hlq.SCIUSMP1.OUT(CIUMIGCD) job.

Migrating application definitions

To migrate the application definitions that you defined in CICS IA Version 3.1 to CICS IA Version 3.2, refer to the DB2 tables migration.

See “Migrating the DB2 tables” on page 60.

Defining resources to CICS

Jobs and resource definitions are supplied to define CICS IA resources for all CICS releases.

The supplied jobs and resource definitions provide JCL and statements to define the following resources:

- CICS IA program components
- CICS IA transactions CINT and CINB
- VSAM files CIUCNTL, CIUINT1 through CIUINT7, and CIUAFF1 through CIUAFF3
- DB2 entry (DB2ENTRY) definitions and DB2 transaction (DB2TRAN) definitions
- CINT transient data queue for messages
- Atomservice resources required for the CICS controlling region (CICS TS V4.1 or above) used to manage data collection from the CICS Explorer plug-in
- LIBRARY resources (available for CICS IA V3.2 and above)

To define the CICS resources for the CICS IA collector, edit and run job CIUJCINT.

Note: In this job, you can also define LIBRARY resources for the SCIULOAD and SCIULODE/K data sets, and they will be loaded dynamically during the startup job execution.

To define the CICS Atomservice resources, edit and run job CIUJWEB2.

To define the CICS file resources for the CICS IA collector, edit and run one or both of the following jobs:

- CIUJCFIL to define the CICS file resources locally or in a file-owning region
- CIUJCFIR to define the CICS file resources as remote to a file-owning region

To define the CICS IA national language message handler and panels to CICS, use the definition supplied in hlq.SCIUSAME . This version is the default English version, and you must always use it, unless you require the Japanese version. For the Japanese version, use the definition supplied in hlq.SCIUSAMK.

You create the names of the CICS IA resource definition groups during configuration.

The group names of CICS resource definitions are shown in Table 7. For the information on the definitions contained in these groups, see also hlq.SCIUSAME and hlq.SCIUSAMK.

Table 7. The groups of CICS resource definitions

Group name	Description	Contained definition(s)
<i>_groupf_</i>	Group name for local files	CIUDEFB
<i>_groupr_</i>	Group name for remote files	CIUDEFR
<i>_group_</i>	Group name for CINT CSD and LIBRARY resource definitions	CIUDEFB CIUDEFTE CIUDEFLL CIUDEFLE
<i>_groupw_</i>	Group name for Atomservice definitions	CIUDEFW

Installing resource definitions

To install CICS IA resource definitions in every region, in which CICS IA will run, use one of two methods.

About this task

1. Add the CICS IA resource group to the group list specified in the startup JCL for your CICS region.

Note that a definition for a DB2 connection is not supplied in the CICS IA resource definition. CICS IA defines a DB2 entry for CICSIA and a number of DB2 transaction definitions. The CICS IA definition group must come after the customer-created resource group that contains the corresponding DB2CONN connection definition. Otherwise, an SQL -923 error occurs when any of the CICS IA plans are run.

2. Use the CEDA transaction to install the CICS IA group in your running system.

Collecting DB2 resources in your CICS region

You must edit the JCL to conform with your own system conventions.

About this task

Before collecting DB2 resource information in your CICS region, it is highly recommended that you create a new index against your SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT tables. This will improve performance of the online collection. If you choose not to create these indexes, switch off the DB2 query in the SYSIBM tables by entering N in the Inquire for DB2 Resources field in the CINT General Options panel for that region. See “Specifying region-specific options: general” on page 93 for details.

To collect DB2 data:

1. Review the following members in hlq.SCIUSMP2.OUT:
 - CIUDBCT
 - CIUDBNT
2. Review the associate SQL members for the above jobs. They can be found in hlq.SCIUSQL.OUT.

3. Run `hlq.SCIUSMP2.OUT(CIUDBCT)` to create indexes for SYSIBM.
4. Run `hlq.SCIUSMP2.OUT(CIUDBNT)` to bind the programs that update DB2 tables.

CICS IA Natural support

With CICS IA, you can gather CICS resource information for COBOL, PL/I, and Assembler language programs called from within a Software AG Natural program.

Additionally, with CICS IA you can gather detailed information on both Adabas calls and Natural program calls from Natural applications running in the Software AG Natural environment. In this way, you can identify Adabas resources used by Natural applications and Natural program relationships. You can use the Natural Resource Options to control the collection data on Adabas calls and Natural program calls as described in the “Changing the Collector options” on page 90.

CICS IA Natural support runs as an exit to the Natural Review Data Collector. Both the Natural Review Data Collector and the CICS IA Natural support exit must be link-edited into a Natural Shared Nucleus (NSN). The CICS IA Natural support exit is a TP-specific module for the CICS environment, and must be included as part of a Single-Environment Shared Nucleus for CICS. Refer to the *Natural Operations Manual* for more information regarding the Natural Shared Nucleus, TP-specific modules, and Single-Environment Shared Nuclei.

The installation of Natural Support will not change the structure of the data interfaces to CICS IA. However, the content of the data collected will change in that the name of the Natural nucleus will be replaced by the name of the Natural program that is currently executing.

Installing Natural support

CICS IA Natural support runs as an exit to the Natural Review Data Collector. The Natural Review Data Collector and the CICS IA Natural support exit must be link-edited into an NSN.

Before you begin

Follow these steps to install Natural Support:

Procedure

1. Verify that the CICS IA Natural exit is available, using program `CIURDCX1` found in the `hlq.SCIULOAD` library
2. Follow the instructions for link-editing the Natural Review Data Collector, (`NATRDC`) into an NSN as provided by the vendor. The exit can be linked into one of three places:
 - The Independent (shared) nucleus `NATvvvSH`.
 - The CICS dependent nucleus `NCvvvRE`.
 - A dynamically loaded CICS parameter module.
3. Link the exit into the CICS dependent nucleus, `NCvvvRE`:
 - a. Specify the `RDCEXIT` and `RDCSIZE` parameters in the Natural/CICS parameter module. These parameters are options on the `NTPRM` macro. Set the parameters as follows:
 - `RDCSIZE=2`
 - `RDCEXIT=(CIURDCX1,400)`

4. Add the following DD statement for the CICS IA load library to the job used to link-edit the NSN:

```
//ACIUMOD DD DISP=SHR,DSN=hlq.ACIUMOD
```

5. Add the following linkage-editor control statement to the linkage-editor input stream:

```
INCLUDE ACIUMOD(CIURDCX1)
```

6. If you are linking into the Independent (shared) nucleus add the following into the job stream:

```
//SCIULOAD DD DISP=SHR,DSN=hlq.SCIULOAD
```

```
INCLUDE SCIULOAD(CIURDCX1)
```

What to do next

When the above steps have been completed the CICS IA collection of interdependency data will record the names of the Natural programs involved in place of the name of the Natural nucleus.

Customizing the CICS IA Natural Interface

Customizing the CICS IA Natural Interface includes two tasks: Modifying the Exclusive Work Area (EWA) size and the Natural Name List size.

Modifying the Exclusive Work Area (EWA) size

The default size of the EWA allows you to collect information for up to 12 Natural programs for each Natural session. You can change the EWA size by modifying the RDCEXIT Natural profile parameter for the CICS IA Natural support exit. If you increase the EWA by 8 bytes you can collect information for one more Natural program.

Modifying the Natural Name List size

The CICS IA Natural support exit extracts the Natural information and saves it in an area called the Natural Name List (NNL). The default size of the NNL allows you to collect information on up to 252 simultaneous Natural sessions.

To modify the size of the NNL area:

1. Update the supplied source member hlq.SCIUSRCE(CIUNNLP). Follow the instructions in the source.

```
* NSESS specifies the maximum number of simultaneous Natural
* sessions. The number must be a value between 2 and 32256.
* A value 252 is the installation default setting.
*
CIUNNL  DS    0D                                Natural Name list
        CIUNNLD DSECT=NO,NSESS=252
        SPACE
        END
```

2. Review and modify sample job hlq.SCIUSMP1.OUT(CIUJCNNL):
 - Modify the Job card to meet your site standards.
 - Review the ASM.SYSIN card to make sure you are picking up the modified source.
 - Review the LKED.SYSLMOD card to make sure you are linking the module into the required dataset.

3. To make the modified NNL table available to the IA Natural Collector, ensure that the dataset into which you have linked the modified NNL module is before the CICS IA load library in the DFHRPL.

Security considerations

Security considerations are described in an appendix.

For information on how to set up RACF security for CICS IA refer to Appendix G, "CICS IA security," on page 345.

Tailoring your CICS startup job

To enable CICS IA to run in your CICS region, do the following when setting up your CICS startup job:

About this task

1. 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.
2. If you use VSAM RLS to share the dependency data file and control record file across multiple regions, specify the system initialization parameter RLS=YES.
3. Add the following load libraries to the DFHRPL concatenation in the startup job JCL:
 - hlq.SCIULOAD
 - hlq.SCIULODE (the default, English, national language: this is always required)
 - Optionally, add hlq.SCIULODJ (if you require Japanese national language support)

Instead of adding the load libraries to DFHRPL, you can edit the CIUJCINT job to define LIBRARY resources for the SCIULOAD and SCIULODE/K data sets, and they will be loaded dynamically during the startup job execution.

Note: Available for CICS IA V3.2 or above.

4. Add the following DD statement for the CINT transient data message log:

```
//CINTLOG DD SYSOUT=*
```
5. On any region where you intend to collect DB2 data, ensure that the user ID, under which CICS IA runs, has GRANT permission to the batch plan created in the sample job SCIUSMP2(CIUDBNT). This will enable the background transaction CINB to access the SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT DB2 tables.

Starting and stopping CICS IA from the PLT

You can start CICS IA from a program list table (PLT) program initiated during the third stage of CICS initialization; that is, a program specified in the second part of the program list table post initialization (PLTPI) list for the CICS region.

About this task

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

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

A program list table shutdown (PLTSD) program to stop the Collector, CIUSTOP, is supplied with CICS IA. To stop CICS IA from the PLT:

1. Add the following to your PLT shutdown table:
DFHPLT TYPE=ENTRY,PROGRAM=CIUSTOP
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
2. Ensure that the definition of the CINT transaction specifies SHUTDOWN(ENABLED).

For definitive information about installing and running PLTPI and PLTSD programs, see the *CICS Customization Guide*.

Restarting your CICS regions

Restart each CICS region in which CICS IA is to run, using a CICS startup job modified for CICS IA support.

How to restart your CICS regions is described in “Tailoring your CICS startup job” on page 65.

CICS IA supplied modules required in the MVS linklist

If you use the MVS interactive problem control system (IPCS) to format and analyze CICS system dumps, you can use the CICS IA system dump formatting routines, CIUIADUF and CIUICDUF, to format the CICS IA collector data areas.

To make these routines accessible to IPCS, copy them from hlq.SCIULOAD into a data set of your choice that is in the MVS linklist.

Chapter 4. Running the installation verification program

Use the CICS IA installation verification program (IVP) to verify that CICS IA has been installed correctly on your system.

You are strongly recommended to fix any errors reported by the IVP before running any other CICS IA program.

The rest of this section contains:

- “About the IVP”
- “Running the IVP”
- “Loading IVP sample data” on page 68
- “Viewing the IVP sample data” on page 68

About the IVP

The installation verification program checks CICS RDO definitions to ensure that all software elements, programs, maps, transactions, files, transient data queues, and DB2 entries are correctly defined to CICS and are available.

During the verification process, the IVP writes messages to the CICS system log. These messages indicate the success or failure of the installation. They show verified elements and also any elements of CICS IA that are missing or unavailable. The lack or unavailability of an element could be caused, for example, by an error in loading the software, or by a faulty RDO definition.

The messages that can be issued by the IVP are in the range CIU1001 through CIU1013. They are listed in “Messages that CICS IA can issue” on page 255. Error messages give the probable cause of the error and the action to be taken to correct it.

Running the IVP

How to run the IVP.

About this task

1. Clear the CICS screen.
2. Type the transaction name CIUT.
3. Press Enter to start the program. The program might take several minutes to validate the software.
4. When the verification program has completed, check the messages displayed on your terminal:

CICS IA was installed successfully

If the IVP finds that CICS IA was installed successfully, the following message appears:

```
CIU1002I  INSTALLATION VERIFICATION ENDED SUCCESSFULLY
```

Quit the IVP and go to “Loading IVP sample data” on page 68.

CICS IA was not installed successfully

If the IVP finds that one or more elements of CICS IA were not installed correctly, or are missing, the following message appears:

```
CIU1009 VERIFICATION UNSUCCESSFUL - HIGHEST RETURN CODE: n
```

Go to step 5.

5. Check all the messages that were issued by the IVP during the verification process. Messages indicating either successful or unsuccessful verification of each CICS IA software element are written to the CICS system log. The default CICS system log is CSMT.

Messages issued by the IVP are in the range CIU1001 through CIU1013 and are listed in “Messages that CICS IA can issue” on page 255.

6. Locate any missing resources identified by the IVP. Ensure that all the resources required by CICS IA are correctly defined to CICS and are available.

If you cannot locate or restore a missing resource, contact the IBM Software Support Center (ISC).

7. Run the IVP again until it confirms that CICS IA has been installed correctly.

Loading IVP sample data

When the IVP confirms that CICS IA has been 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.

About this task

Edit and run the CIUIVPLD and CIUALOAD jobs to load the sample IVP data.

The CIUIVPLD and CIUALOAD jobs are put into the hlq.SCIUSMP2 library by the CICS IA installation procedure where “hlq” is a prefix defined during installation. Before running CIUIVPLD and CIUALOAD edit them to meet the requirements of your system.

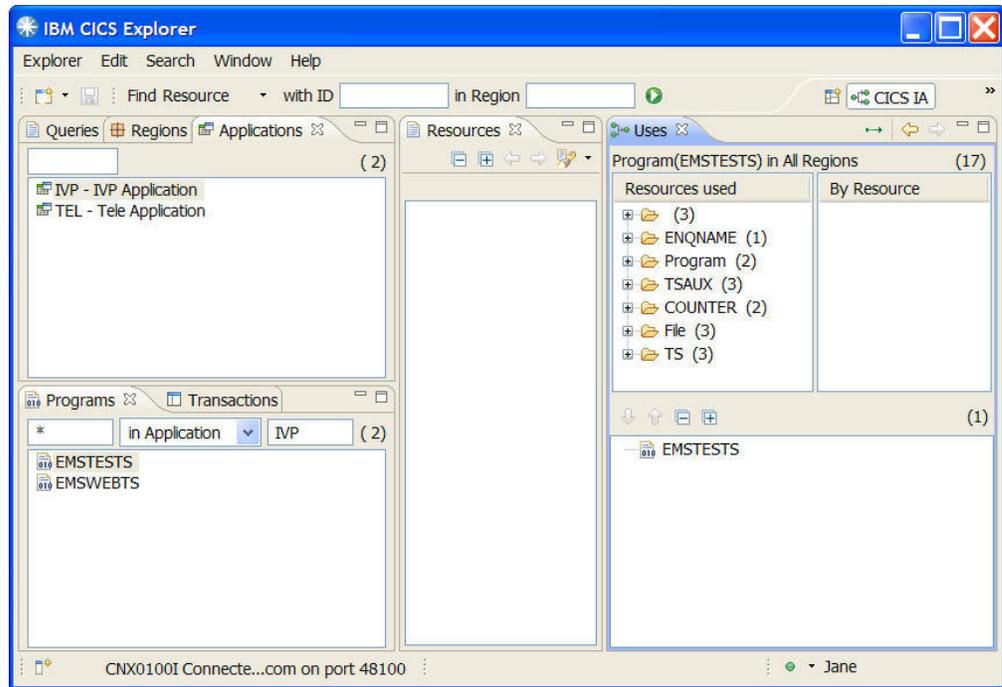
After you have run the CIUIVPLD and CIUALOAD jobs successfully, you can use the CICS IA Explorer plug-in to view the sample data. For detailed instructions on how to view and analyze data with the CICS Explorer, see *IBM CICS Explorer User Guide*.

Viewing the IVP sample data

After you have run the CIUIVPLD job successfully, you can use the CICS IA plug-in interface to interrogate the sample data.

About this task

1. Right-click IVP - IVP Application in the **Applications** view.
2. Select **Programs Used**, the results are displayed in the **Programs** view.
3. Right-click EMSTESTS in the **Programs** view and select **Uses Resources>All Regions**, the results are displayed in the **Uses** view.



Chapter 5. Running the Collector

You run the CICS IA Collector to look for instances of program commands that might cause resource dependencies or transaction affinities.

This section contains the following topics:

- “Displaying the Collector Main Administration Menu panel” on page 72
- “Controlling the collection of dependency and affinity data” on page 80, which contains:
 - “Starting the collection of data” on page 81
 - “Changing the data collection options dynamically” on page 84
 - “Pausing the collection of data” on page 85
 - “Resuming the collection of data” on page 86
 - “Stopping the collection of data” on page 87
 - “Displaying Collector statistics for a specified region” on page 89
- “Changing the Collector options” on page 90
- “Collector errors” on page 119
- “Controlling Collector performance” on page 355

The CICS and non-CICS commands that the Collector can monitor are listed in “Dependency-related commands” on page 6.

You can run the Collector either at a CICS 3270-type terminal (through interactive screens or single-line commands), from a console, or from an application program. This section primarily describes how to use the Collector through the interactive screens at a CICS terminal, but also gives equivalent commands to use at a terminal, at a console, or in an application program.

For an overview of the Collector, see “The Collector component” on page 15.

To control the Collector:

- Change its state, which is described in “Controlling the collection of dependency and affinity data” on page 80.
- Change its run time options, which is described in “Changing the Collector options” on page 90.

Running the Collector for the first time

To run the Collector for the first time configure the region options using option 2 of the Collector Main Administration Screen before gathering any dependency or affinity data.

See Figure 15 on page 72, to:

- Specify at least one CICS region to be monitored
- Specify (in further screens) the CICS, DB2, MQ, and IMS commands to be monitored on that region
- Specify (in a further screen) whether dependency or affinity data is to be collected on that region, with some other region-specific options

For more information about setting Collector options, see “Changing the Collector options” on page 90.

If you have migrated the CIUCNTL control file from a previous CICS IA release, some CICS regions might already be configured for monitoring. For information about migrating the CICS IA control file, see “Migrating the VSAM control record file” on page 61

If you have not migrated the CIUCNTL control file from a previous CICS IA release (in which case, CICS IA's global values will already have been set up), before you try to gather any dependency or affinity data choose the global values menu options to set up CICS IA's global values, see Figure 35 on page 114.

Displaying the Collector Main Administration Menu panel

Display the Collector Main Administration Menu panel by following the steps below.

About this task

To display the Collector Main Administration Menu panel:

1. At a CICS terminal type the transaction identifier CINT
2. Press Enter and panel CIU000, see Collector Main Administration Menu screen, CIU000, is displayed. You can use the Main Administration Menu panel to:
 - Display the Operations Menu panel, which enables you to review and change the state of the Collector, on each of the CICS regions being monitored.
 - Display configuration panels that enable you to set the run time options used by the Collector.
3. Press the F3 (or F12) function key to close the Collector Main Administration Menu panel. Closing the panel does not affect the state of the Collector.

```
CIU000          CICS Interdependency Analyzer for z/OS - V3R2M0          2011/01/01
                Main Administration Menu                                09:25:50AM

Select one of the following. Then press Enter.  0

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

CICS Sysid: TLS3  CICS Applid: IYCLZC03  TermID: TC20  5
CIU7000I 5697-J23 (C) Copyright IBM Corp. 2001, 2011  6
F1=Help   F2=          F3=Exit   F4=          F5=          F6=  7
F7=       F8=          F9=       F10=         F11=         F12= Exit
```

Figure 15. Collector Main Administration Menu panel, CIU000

The meaning of each part of the Collector Main Administration Menu panel, CIU000, is as follows:

- 0 The functions that you can select from this state of the Collector. For any state of the Collector, only appropriate functions are displayed. Type a number from 1 through 3 at the cursor. Then press Enter.

1 Displays the Operations Menu panel, shown in Collector Operations Menu screen, CIU100, which enables you to review the current state of the Collector on each of the CICS regions being monitored by CICS IA, to start, stop, pause, or resume the Collector on any of those regions, and to display statistics about any of the regions.

2 Displays the Region Configuration Menu panel, shown in Collector Region Configuration Menu screen, CIU200, which enables you to specify the CICS regions that are to be monitored by CICS IA and, in further panels, to specify which CICS, DB2, IMS, and MQ commands are to be monitored.

3 Displays the Global Configuration Menu panel, shown in Collector Global Options Menu screen, CIU300, which enables you to specify global values for CICS IA, such as national language, date and time formats, trace level and HLQ for dump data set.

4 Displays the User Administration Menu panel, CIU400, with which you can add, delete or display user records for the CICS IA Command Flow feature.

5 The 4-character system ID (SYSID) and APPLID of the CICS region on which the CINT transaction is running, with the terminal identifier (TERMIN) of the terminal from which it was started.

6 The message line used to display diagnostic messages. When the CINT transaction is first entered, this line displays the copyright notice:

CIU7000I 5697-J23 (C) Copyright IBM Corp. 2001, 2011

7 The keys that select functions to affect the operation of the Collector or to get help information about it. This line displays all possible functions, not all of which are appropriate, or selectable, for a given state of the Collector.

The CINC transaction User Administration

With the CINC transaction User Administration feature you can view and manage the accounts of the Command Flow collector users. This functionality is implemented through the CINT transaction and provides the usage of the Command Flow collector for multiple regions.

Managing Command Flow collector users

With the CICS IA User Administration Menu panel you can manage the Command Flow collector users.

About this task

You can use the User Administration Menu panel to add, copy, delete the Command Flow collector users or see specific information about users.

Procedure

1. At a CICS terminal, type the transaction identifier CINT.
2. Press Enter. The Collector Main Administration Menu panel, CIU000, is displayed. See Figure 15 on page 72.
3. On panel CIU000, choose option 4.
4. Press Enter. The User Administration Menu panel, CIU400, is displayed. See Figure 16 on page 74.

```

CIU400      CICS Interdependency Analyser for Z/OS - V3R2M0      2011/03/02
              User Administration Menu                          12:15:07PM

Type action code then press Enter                               Page : 1 of 1
1=Add User  2=Copy User  3=Delete User  4=User Details  1

   CINC      CINC      CINC      CINC
Act User ID  USER STATUS  Act User ID  USER STATUS
--- SSUSERSS  NOT ACTIVE  ---
--- ##USER##  NOT ACTIVE  ---
--- @@USER@@  NOT ACTIVE  ---
--- TTTTTTTT  NOT ACTIVE  ---
--- USER1     NOT ACTIVE  ---
--- XXUSERXX  NOT ACTIVE  ---
--- 1USER1    NOT ACTIVE  ---
--- 33333333  ACTIVE       ---
---

CICS Sysid: T41B      CICS Applid: IYDZT41B      TermID: TC47

F1=Help      F2=          F3=Exit    F4=      F5=Refresh  F6= 2
F7=Page Up   F8=Page Down F9=          F10=     F11=       F12=

```

Figure 16. User Administration Menu panel, CIU400

The meaning of each part of the CICS IA User Administration Menu panel, CIU400, is as follows:

1 Specify one of the following action codes:

1=Add User: In any of the Act fields, type action code 1 and press Enter. The Add User Menu panel, CIU410, is displayed.

2=Copy User: In the Act field corresponding to the user that you want to copy, type action code 2 and press Enter. The Copy User Menu panel, CIU420, is displayed.

3=Delete User: In the Act field corresponding to the user that you want to delete, type action code 3 and press Enter. You are asked to confirm that the specified user is to be deleted.

Note: To delete a user, make sure that the Command Flow collector is stopped and the CINC session for this user is closed.

4=User Details: In the Act field corresponding to the chosen user, type action code 4. The User Details Menu panel, CIU440, is displayed.

2 The control keys that you can use to select functions that control the operation of the User Administration feature and provide help information about it. Actions are summarized in Table 8.

Table 8. The control keys on the User Administration Menu panel

Action	Function key
Return to the Main Administration Menu panel, CIU000.	F3
Sort the list of the users and refresh the Command Flow collector status for the users.	F5

Table 8. The control keys on the User Administration Menu panel (continued)

Action	Function key
Scroll the list of the Command Flow collector users up.	F7
Scroll the list of the Command Flow collector users down.	F8
Cancel the deletion of a user.	F12

5. Press F3 to close the CICS IA User Administration Menu panel.

Adding new Command Flow collector users

With the CICS IA Add User Menu panel you can add new users to the list of the Command Flow collector users.

About this task

You can use the Add User Menu panel to add new Command Flow collector users.

Procedure

1. On the User Administration Menu panel, CIU400, type action code 1.
2. Press Enter. The Add User Menu panel, CIU410, is displayed; see Figure 17.

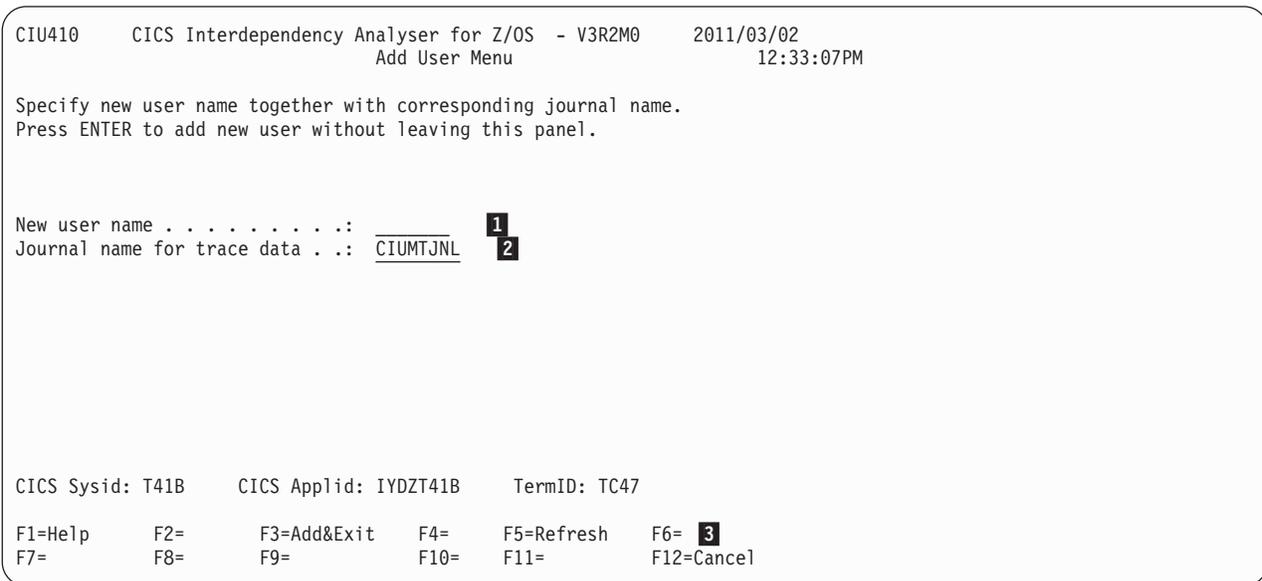


Figure 17. Add User Menu panel, CIU410

The meaning of each part of the CICS IA Add User Menu panel, CIU410, is as follows:

- 1** Specify the name of the CINC user that you want to add.
- 2** Specify the journal name for trace data. The length of the name is up to 8 characters. The default journal name is CIUMTJNL.

Note:

- a. The journal name must have the same name as the corresponding journal model.
- b. The log stream name in a given journal model must be the same on all regions.

3 The control keys that you can use to select functions that control the operation of the CICS IA User Administration feature and provide help information about it. Actions are summarized in Table 9.

Table 9. The control keys on the Add User Menu panel

Action	Function key
Add a new user and return to the User Administration Menu panel, CIU400.	F3
Clear the New user name and Journal name for trace data fields.	F5
Add a new user without leaving the panel.	Enter
Return to the User Administration Menu panel without adding the new user specified on the Add User Menu panel.	F12

- 3. Press F3 (or F12) to close the CICS IA Add User Menu panel.

Copying Command Flow collector users

With the CICS IA Copy User Menu panel you can copy the Command Flow collector users.

About this task

You can use the CICS IA Copy User Menu panel to create a new Command Flow collector user by copying the parameters of an existing user.

Procedure

1. On the User Administration Menu panel, CIU400, type action code 2.
2. Press Enter. The Copy User Menu panel, CIU420, is displayed; see Figure 18 on page 77.

```

CIU420      CICS Interdependency Analyser for Z/OS - V3R2M0      2011/03/02
              Copy User Menu                                     12:38:07PM

Press Enter to add new user without leaving this panel.

Specify parameters for the copy of user XXUSERXX :
New user name . . . . . : USER1313      1
Journal name for trace data . . : CIUMTJNL      2

Parameters below will be copied
Traced transactions IDs. . . . . :      3
Traced user USERID . . . . . : XXUSERXX      4
Traced terminal TERMID . . . . . : *      5
Command Flow data ID . . . . . :      6
Dynamic call . . . . . : Y      7
User modifiable exit name. . . . . :      8
Traced regions APPLIDs . . . . . :      9

CICS Sysid: T41B      CICS Applid: IYDZT41B      TermID: TC47
CIU2514I User USER1313 was added.
F1=Help      F2=      F3=Copy&Exit      F4=      F5=Refresh      F6= 10
F7=      F8=      F9=      F10=      F11=      F12=Cancel

```

Figure 18. Copy User Menu panel, CIU420

The meaning of each part of the CICS IA Copy User Menu panel, CIU420, is as follows:

1 The 8-character identifier for a copy of user that you want to create.

2 Specify the journal name for trace data. The length of the name is up to 8 characters. The default journal name is CIUMTJNL.

Note:

- a. The journal name must have the same name as the corresponding journal model.
- b. The log stream name in a given journal model must be the same on all regions.

3 The identifiers of the transactions for which you want to trace the Command Flow data.

4 The user identifier associated with CICS transactions, for which you want to collect the Command Flow data.

5 The terminal identifier associated with CICS transactions, for which you want to collect the Command Flow data.

6 The name of the Command Flow trace that is to be captured.

7 A flag that determines whether or not dynamic call programs are captured. Specify Y (Yes) or N (No). The default value of this field is Y.

8 The 8-character user modifiable exit name that you can use to add data to a user Command Flow record.

| **9** The 8-character APPLID of the CICS region for which you want to trace the
| Command Flow data.

| **10** The control keys that you can use to select functions that control the
| operation of the User Administration feature and provide help information about
| it. Actions are summarized in Table 10.

| *Table 10. The control keys on the Copy User Menu panel*

Action	Function key
Copy a user and return to the User Administration Menu panel, CIU400.	F3
Clear the user parameters fields on panel CIU420.	F5
Copy a user without leaving the panel.	Enter
Return to the User Administration Menu panel, CIU400.	F12

| 3. Press F3 (or F12) to close the CICS IA Copy User Menu panel.

| **Viewing details of a Command Flow collector user**

| With the CICS IA User Details Menu panel you can see collection options and
| statistics information for the chosen user.

| **About this task**

| You can use the CICS IA User Details Menu panel to display a Command Flow
| collector user statistics information.

| **Procedure**

- | 1. On the User Administration Menu panel, CIU400, type action code 4.
- | 2. Press Enter. The User Details Menu panel, CIU440 is displayed; see Figure 19
| on page 79.

```

CIU440      CICS Interdependency Analyser for Z/OS - V3R2M0      2011/03/14
              User Details Menu                                02:19:07PM

Options of CINC user DSFSDFFS
Traced transactions IDs. . . . . 1
Traced user USERID . . . . . DSFSDFFS 2
Traced terminal TERMID . . . . . * 3
Command flow data ID . . . . . 4
Journal name for trace data. . . . . CIUMTJNL 5
Dynamic call for trace . . . . . Y 6
User modifiable exit name. . . . . 7
Traced region APPLIDs. . . . . 8

Last data collection statistics

Collector status . . . . . NOT ACTIVE 9
Collector last start . . . . . 10
Collector last stop. . . . . 11

CICS Sysid: T41B      CICS Applid: IYDZT41B      TermID: TC60

F1=Help      F2=      F3=Save&Exit      F4=      F5=Refresh      F6= 12
F7=          F8=          F9=          F10=     F11=          F12=Cancel

```

Figure 19. User Details Menu panel, CIU440

The meaning of each part of the CICS IA User Details Menu panel, CIU440, is as follows:

- 1 The identifiers of the transactions for which you want to trace the Command Flow data.
- 2 The user identifier associated with CICS transactions, for which you want to collect the Command Flow data.
- 3 The terminal identifier associated with CICS transactions, for which you want to collect the Command Flow data.
- 4 The name of the Command Flow trace that is to be captured.
- 2 Specify the journal name for trace data. The length of the name is up to 8 characters. The default journal name is CIUMTJNL.

Note:

- a. The journal name must have the same name as the corresponding journal model.
- b. The log stream name in a given journal model must be the same on all regions.
- 6 A flag that determines whether or not dynamic call programs are captured. Specify Y (Yes) or N (No). The default value of this field is Y.
- 7 The 8-character user modifiable exit name that you can use to add data to a user Command Flow records.
- 8 The 8-character APPLID of the CICS region for which you want to trace the Command Flow data.

9 The Command Flow collector status for the specified user.

10 The date and time when the Command Flow data collection process was last started. The time shown is the local time, and the date is given in the format specified by the Global Options Menu panel of the CINT transaction.

11 The date and time when the Command Flow data collection process was last stopped. The time shown is the local time, and the date is given in the format specified by the Global Options Menu panel of the CINT transaction.

12 The control keys that you can use to select functions that control the operation of the CINC collector and provide help information about it. Actions are summarized in Table 11.

Table 11. The control keys on the User Details Menu panel

Action	Function key
Save all the changes and return to the User Administration Menu panel, CIU400.	F3
Refresh the CINC user option fields.	F5
Update the journal name without leaving panel CIU440.	Enter
Return to the User Administration Menu panel, CIU400.	F12

3. Press F3 (or F12) to close the CICS IA User Details Menu panel.

Controlling the collection of dependency and affinity data

Use the menu CIU100 to control the collection of dependency and affinity data.

Select **1 Operations Menu** from the Collector Main Administration Menu screen, CIU000 (shown in Figure 15 on page 72). The Collector Operations Menu screen, CIU100, shown in Figure 20 on page 81, is displayed. From this screen, you can perform these actions:

- Review the current state of the Collector on each of the CICS regions being monitored by CICS IA.
- Start, stop, pause, or restart the Collector on any or all of the monitored regions.
- Call up a further screen to show Collector statistics for any of the monitored regions.
- Refresh the Collector options on any or all of the monitored regions.

```

CIU100          CICS Interdependency Analyzer for z/OS - V3R1M0      2007/09/26
                  Operations Menu                                     09:25:50AM

Type action code then press ENTER.                               More : +

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

  CICS      CICS      Start      Start
Act Applid  Sysid Status  Date       Time       Collecting
-   ALL     ALL          -          -          1
-   IYCLZC03 TLS3  RUNNING   2007/09/25 04:56:07PM Dependencies
-   IYCLZC04 TLS4  STOPPED
-   IYCLZC05 TLS5  RUNNING   2007/09/26 08:59:23AM Affinities
-
-
-
-
-
-
-
-
-
-
-

CICS Sysid:  TLS3   CICS Applid:  IYCLZC03   TermID:  TC20

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

```

Figure 20. Collector Operations Menu screen, CIU100

1 If more than one CICS region is listed on the Operations Menu, an extra item, with an APPLID and SYSID of “ALL”, is displayed at the top of the list. With this item you can specify that the chosen operation is to be applied to all the regions in the list. You can select ALL with all action codes except 5, Statistics.

Starting the collection of data

You can start collecting dependency data, affinity data or both on a particular region only when the Collector is stopped on that region.

To start collecting data on a specified CICS region, use one of the methods shown in Methods for starting data collection by the Collector.

Table 12. Methods for starting data collection by the Collector

Where used	Command or function key
CINT transaction Operations Menu, CIU100	Type 1 against a listed CICS APPLID and press Enter, to start the Collector on the specified region. 1
CINT transaction Operations Menu, CIU100	If more than one CICS region is listed on the Operations Menu, you can type 1 against the first CICS APPLID in the list (“ALL”) and press Enter to start the Collector on all the regions. 2
3270 terminal	CINT <start_options> 3
Console	F cicsjob, CINT <start_options> 4
Application program	EXEC CICS START TRANSID('CINT') FROM('<start_options>') 5

Notes:

1 If you enter action code 1 against a CICS region, you are asked to confirm that you want the Collector to start recording data.

2 If you enter action code 1 against ALL, you are asked to confirm that you want the Collector to start recording data on all the regions.

3 The start options are:

START

Starts the Collector on the local CICS region; that is, the region to which the 3270 terminal is connected. The type of data collected (interdependency or affinity) depends on what you have specified for the Data to Collect region-specific option on the Collector General Options screen, CIU260, shown in Collector General Options screen, CIU260.

STARTALL

Starts the Collectors on all the monitored CICS regions. For each region, the type of data collected depends on what you have specified for the Data to Collect region-specific option on the Collector General Options screen, CIU260.

STARTAFF

Starts the collection of affinity data on the local CICS region. This command overrides what you have specified for the Data to Collect region-specific option. In other words, even if you have specified the type of Data to Collect as interdependency, the CINT STARTAFF command causes affinity, not interdependency, data to be collected.

STARTALLAFF

Starts the collection of affinity data on all the monitored CICS regions. This command overrides what you have specified, for each region, on the Data to Collect region-specific option. In other words, even if, for some regions, you have specified the type of Data to Collect as interdependency, the CINT STARTALLAFF command causes affinity data to be collected on all the regions.

STARTINT

Starts the collection of interdependency data on the local CICS region. This command overrides what you have specified for the Data to Collect region-specific option. In other words, even if you have specified the type of Data to Collect as affinity, the CINT STARTINT command causes interdependency, not affinity, data to be collected.

STARTALLINT

Starts the collection of interdependency data on all the monitored CICS regions. This command overrides what you have specified, for each region, on the Data to Collect region-specific option. In other words, even if, for some regions, you have specified the type of Data to Collect as affinity, the CINT STARTALLINT command causes interdependency data to be collected on all the regions.

STARTBOTH

Starts the collection of both affinity data and interdependency data on the local CICS region. This command overrides what you have specified for the Data to Collect region-specific option. In other words, even if you have specified the type of Data to Collect as affinity, the CINT STARTBOTH command causes both affinity data and interdependency data to be collected.

STARTALLBOTH

Starts the collection of both affinity data and interdependency data on all the monitored CICS regions. This command overrides what you have specified for the Data to Collect option in each region. In other words,

even if you have specified, for each region, the type of Data to Collect as affinity, the CINT STARTALLBOTH command causes both affinity data and interdependency data to be collected for all the regions.

4 cicsjob is the name of your CICS startup job. The <startup_options> are as described for the CINT command issued from a 3270 terminal, under **3** above.

5 This command, issued from a program initiated during the third stage of CICS initialization (that is, a program specified in the second part of the PLTPI list for the CICS region), starts the Collector during CICS initialization. The <startup_options> are as described under **3** above. A PLTPI program to start the Collector, CIUSTART, is supplied with CICS IA. For more information, see Starting and stopping CICS IA from the PLT.

Using one of the methods listed in Methods for starting data collection by the Collector causes the Collector to record transaction dependencies or affinities, until you pause or stop data collection.

On each region, the data that is collected depends on:

1. The start option (START, STARTAFF, STARTINT, STARTALLAFF, and so on) that you specify, as noted above.
2. The region-specific options that you specify. For example, if you are collecting dependency data, data is collected only for the (CICS and non-CICS) API commands that you choose to be monitored (by specifying Y for the command type on the CICS Resources Options screen, CIU240, and the DB2/MQ/IMS/RMI Trace Resource Options screen, CIU250. See Collector CICS Resources Options screen, CIU240 and Collector DB2/MQ/IMS Resource Options screen, CIU250).

Similarly, if you are collecting affinity data, data is collected only for the CICS API commands that you choose to be monitored (by specifying Y for the command type on the CICS Affinities Options screen, CIU270. See Collector Affinities Options screen, CIU270).

In addition, data collection is filtered by the region-specific options that you set on the General Options screen, CIU260. See "Transid prefix", "Program exclude list", and "Transaction exclude list", under Specifying region-specific options: general.

For a complete list of the program commands that are not monitored, see What is not monitored.

Each time the Collector is started, a new data space is created. You specify its size on the Collector's Regional Resource Options screen, CIU260—see Collector General Options screen, CIU260. You can also specify that data from the VSAM dependency or affinity files (for example, from previous CICS IA runs) is to be loaded into the data space when it is created. For more information about the Regional Resource Options screen, see Specifying region-specific options.

Note: If there are a large number of data records to be loaded into the data space when it is created (for example, from previous CICS IA runs), the Operations Menu screen might be frozen for some appreciable time, until the records have been loaded.

Changing the data collection options dynamically

You can change the CICS IA monitoring options without restarting CICS IA.

To change which CINT resources to monitor:

1. Change the collector options, see “Changing the Collector options” on page 90.
2. Use one of the methods shown in Methods for changing data collection options to start CINT using the updated options.

Table 13. Methods for changing data collection options

Where used	Command or function key
CINT transaction Operations Menu, CIU100	Type 6 against a listed CICS APPLID and press Enter to have CICS IA use the updated set of options for selecting the resources to monitor.
CINT transaction Operations Menu, CIU100	If more than one CICS region is listed on the Operations Menu, you can type 6 against the first CICS APPLID in the list (ALL) and press Enter to have all CICS IA regions use the updated set of options for selecting the resources to monitor.
3270 terminal	CINT REFRESHOPTIONS 1
3270 terminal	CINT REFRESHALLOPTIONS 2
Console	F cicsjob, CINT REFRESHOPTIONS 3
Console	F cicsjob, CINT REFRESHALLOPTIONS 4
Application program	EXEC CICS START TRANSID('CINT') FROM('REFRESHOPTIONS') 5
Application program	EXEC CICS START TRANSID('CINT') FROM('REFRESHALLOPTIONS') 6

Notes:

1

REFRESHOPTIONS

Causes the Collector to read the options that are currently defined for the local CICS region from the control file. The Collector uses these options to override the ones that are currently in use. The Collector must be in the Running state or the Paused state at the time this option is specified; otherwise, no action is taken.

2

REFRESHALLOPTIONS

Causes the Collector for all regions where it is running to read the options that are currently defined for their local CICS region from the control file. Each collector uses the new options to override the options currently in use.

The options that can be updated by the CINT transaction REFRESHOPTIONS or REFRESHALLOPTIONS commands are the following:

- All CICS options on screens CIU240 and CIU245
- All DB2, IMS, WebSphere MQ, or RMI True options on screen CIU250
- The General Options on screen CIU260 that follow:
 - Data to Collect

- Inquire for DB2 resources
 - Maintain usage counts
 - Perform periodic saves
 - Program exclude list
 - Restore data on start
 - Resource prefix list
 - Transaction exclude list
 - Transid prefix
 - Trigger for CINB start
 - TRUEs Include list
 - Size of dataspace
- ALL Affinity options on screen CIU270
 - The Time and Date Options on screen CIU280

3 cicsjob is the name of your CICS startup job. This job refreshes the Collector options on the local CICS region from its region record in the CICS IA control file.

4 cicsjob is the name of your CICS startup job. This job refreshes the Collector options on all the monitored CICS regions from records in the CICS IA control file.

5 This command, issued from a PLTPI program, refreshes the Collector options on the local CICS region from its region record in the CICS IA control file.

6 This command, issued from a PLTPI program, refreshes the Collector options on all the monitored CICS regions from records in the CICS IA control file.

Pausing the collection of data

You can pause the collection of dependency or affinity data on a particular region only when the Collector is running on that region.

To pause the collection of data on a specified CICS region, use one of the methods shown in Table 14. **1**

Table 14. Methods for pausing data collection by the Collector

Where used	Command or function key
CINT transaction's Operations Menu, CIU100	Type 3 against a listed CICS APPLID and press Enter.
CINT transaction's Operations Menu, CIU100	If there is more than one CICS region listed on the Operations Menu, you can type "3" against the first CICS APPLID in the list ("ALL") and press Enter to pause the Collector on all the regions.
3270 terminal	CINT PAUSE 2
3270 terminal	CINT PAUSEALL 3
Console	F cicsjob, CINT PAUSE 4
Console	F cicsjob, CINT PAUSEALL 5
Application program	EXEC CICS START TRANSID('CINT') FROM('PAUSE') 6
Application program	EXEC CICS START TRANSID('CINT') FROM('PAUSEALL') 7

Notes:

1 If you have set up a timer to control the dates and times at which the Collector runs on the specified region, pausing the Collector by any of the methods in Table 14 on page 85 overrides the action of the timer. However, the timer continues to operate and to pause and resume the collection of data at the specified times. For details of how to set up a timer, see “Specifying region-specific options: timers” on page 103.

2 This command pauses the Collector on the local CICS region; that is, the region to which the 3270 terminal is connected.

3 This command pauses the Collectors on all the monitored CICS regions.

4 `cicsjob` is the name of your CICS startup job. This job pauses the Collector on the local CICS region.

5 `cicsjob` is the name of your CICS startup job. This job pauses the Collector on all the monitored CICS regions.

6 This command, issued from a PLTPI program, pauses the Collector on the local CICS region.

7 This command, issued from a PLTPI program on the local CICS region, pauses the Collectors on all the monitored CICS regions.

Using one of the methods listed in Table 14 on page 85 causes the Collector to stop collecting data until you are ready to resume. The data already collected remains in the data space, and is saved to the dependency data or affinity data VSAM files if that option has been set. (You can specify that data be saved when the Collector is paused on the Regional Resource Options screen, as described in “Specifying region-specific options: general” on page 93.)

Resuming the collection of data

You can resume collecting dependency or affinity data on a particular region only when the Collector is paused on that region.

To resume collecting data on a specified CICS region, use one of the methods shown in Table 15. **1**

Table 15. Methods for resuming data collection by the Collector

Where used	Command or function key
CINT transaction's Operations Menu, CIU100	Type 4 against a listed CICS APPLID and press Enter.
CINT transaction's Operations Menu, CIU100	If there is more than one CICS region listed on the Operations Menu, you can type 4 against the first CICS APPLID in the list ALL and press Enter to resume the collection of data on all the regions.
3270 terminal	CINT CONTINUE 2
3270 terminal	CINT CONTINUEALL 3
Console	F <code>cicsjob</code> , CINT CONTINUE 4
Console	F <code>cicsjob</code> , CINT CONTINUEALL 5

Table 15. Methods for resuming data collection by the Collector (continued)

Where used	Command or function key
Application program	EXEC CICS START TRANSID('CINT') FROM('CONTINUE') 6
Application program	EXEC CICS START TRANSID('CINT') FROM('CONTINUEALL') 7

1 If you have set up a timer to control the dates and times at which the Collector runs on the specified region, resuming the collection of data by any of the methods in Table 15 on page 86 overrides the action of the timer. However, the timer continues to operate and to pause and resume the collection of data at the specified times. For details of how to set up a timer, see “Specifying region-specific options: timers” on page 103.

2 This option causes the Collector to resume on the local CICS region: that is, the region to which the 3270 terminal is connected.

3 This command causes the Collectors on all the monitored CICS regions to resume.

4 `cicsjob` is the name of your CICS startup job. This job causes the Collector to resume on the local CICS region.

5 `cicsjob` is the name of your CICS startup job. This job causes the Collectors on all the monitored CICS regions to resume.

6 This command, issued from a PLTPI program, causes the Collector on the local CICS region to resume.

7 This command, issued from a PLTPI program on the local CICS region, causes the Collectors on all the monitored CICS regions to resume.

Using one of the methods listed in Table 15 on page 86 causes the Collector to resume the collection of transaction dependencies or affinities in the CICS region, until you pause or stop data collection.

Stopping the collection of data

You can stop collecting dependency or affinity data on a particular region only when the Collector is running or paused on that region.

To stop collecting data on a specified CICS region, use one of the methods shown in Table 16.

Table 16. Methods for stopping data collection by the Collector

Where used	Command or function key
CINT transaction's Operations Menu, CIU100	Type 2 against a listed CICS APPLID and press Enter. You are asked to confirm that you want data collection to be stopped on the specified region.

Table 16. Methods for stopping data collection by the Collector (continued)

Where used	Command or function key
CINT transaction's Operations Menu, CIU100	If there is more than one CICS region listed on the Operations Menu, you can type 2 against the first CICS APPLID in the list ALL and press Enter to stop the Collector on all the regions. You are asked to confirm that you want data collection to be stopped on all regions.
3270 terminal	CINT STOP 1
3270 terminal	CINT STOPALL 2
Console	F cicsjob, CINT STOP 3
Console	F cicsjob, CINT STOPALL 4
Application program	EXEC CICS START TRANSID('CINT') FROM('STOP') 5
Application program	EXEC CICS START TRANSID('CINT') FROM('STOPALL') 6

Notes:

1 This option stops the Collector on the local CICS region; that is, the region to which the 3270 terminal is connected.

2 This option stops the Collectors on all the monitored CICS regions.

3 cicsjob is the name of your CICS startup job. This job stops the Collector on the local CICS region.

4 cicsjob is the name of your CICS startup job. This job stops the Collectors on all the monitored CICS regions.

5 This command, issued from a program initiated during the first quiesce stage of CICS shutdown (that is, a program specified in the first half of the PLT for CICS shutdown) stops the Collector on the local region. You are recommended to implement this command, to prevent the Collector delaying CICS shutdown if it is not in the STOPPED state. A PLTSD program to stop the Collector, CIUSTOP, is supplied with CICS IA. For information about installing and running PLTSD programs, see the *CICS Customization Guide*.

6 This command, issued from a PLTSD program on the local CICS region, stops the Collectors on all the monitored CICS regions.

Using one of the methods in Table 16 on page 87 stops the Collector recording any dependency or affinity data in the CICS region until you next start the Collector. Stopping the collector also destroys the data space, and saves the data collected to the VSAM data files.

If there are many data records to be saved, the Operations Menu screen might be frozen for some time, until the records have been saved.

You might want to stop the Collector, on a specified CICS region, when it has detected all dependencies there. You can find out when this has happened from the Collector Statistics Menu screen, CIU150; see “Displaying Collector statistics for a specified region” on page 89. When the Collector has detected all dependencies,

the “Date/time of last change” field changes very infrequently and, if optional periodic saves are performed, the “Records written last save” field is consistently near zero.

Displaying Collector statistics for a specified region

You can display statistics for a specified region with the Collector Statistics Menu panel, CIU150.

From the Collector Operations Menu panel, CIU100 (shown in Figure 20 on page 81), type 5 (the action code for Statistics) against a listed CICS APPLID and press Enter. The Collector Statistics Menu panel, CIU150, shown in Figure 21, is displayed.

```

CIU150          CICS Interdependency Analyzer for z/OS - V3R1M0          2009/08/18
                  Statistics Menu for                                   12:40:17PM

      CICS Sysid   : Z325      CICS Applid   : IYDZZ325

CINT state . . . . . : STOPPED      Collecting Dependencies      1
Records written last save. : 0      2      Number of pauses . . . : 0      5
Total records on file. . . : 17     3      Number of saves. . . . : 0      6
Total command flow records : 0      4

Date/time of last start. . : 2009/08/18 10:24:21AM      7
Date/time of last save . . :
Date/time of last change . :

Total time RUNNING . . . . :          (HHHH:MM:SS)      8
Total time PAUSED. . . . . :          (HHHH:MM:SS)

Table dataspace name . . . :          % full      9

CICS Sysid: Z325   CICS Applid: IYDZZ325   TermID: TC01

F1=Help   F2=      F3=End   F4=      F5=Refresh F6=
F7=      F8=      F9=      F10=     F11=      F12=      10

```

Figure 21. Collector Statistics Menu panel, CIU150

The meaning of each part of the Collector Statistics Menu panel, CIU150, is as follows:

1 The current state of the Collector on this region (RUNNING, PAUSED, or STOPPED) and the type of data (dependency, affinity, or both) being collected, if any.

Notes:

1. When you stop the Collector, the CINT state changes to STOPPED only after the Collector has saved the dependency or affinity data.
2. When you pause the Collector, the CINT state changes to PAUSED *before* the Collector saves the data to ensure that the Collector pauses immediately. After the state has changed to PAUSED, you can refresh the data displayed by pressing Enter.

2 Depending on whether the Collector has been configured to collect dependency data, affinity data, or both on this region, the number of records written to either the dependency VSAM data files, affinity VSAM data files, or both dependency and affinity VSAM data files when data was last saved.

- 3** Depending on whether the Collector has been configured to collect dependency data, affinity data, or both on this region, the total number of records in either the dependency VSAM data files, affinity VSAM data files, or both dependency and affinity VSAM data files.
- 4** The total number of the command flow records written to a User Journal.
- 5** The number of times that the Collector has been paused since the last time that it was started.
- 6** The number of times that data has been saved since the last time the Collector was started.
- 7** The dates and times when the Collector was last started, data was saved, and a change was made to a dependency or affinity table. The times are shown in the local time and the dates are given in the format specified by the Date and Time options of the Collector Global Options Menu panel, CIU300; see Figure 35 on page 114.
- 8** The total times that the Collector has been in the running and paused states since it was last started.
- 9** The name and current percentage occupied of the MVS data space being used. This information is not displayed while in STOPPED state.
- 10** The keys that select functions to affect the operation of the Collector or to get help information about it. This line displays all possible functions, not all of which are appropriate (or selectable) for a given state of the Collector.

Changing the Collector options

You can control how the Collector operates by changing the options that it uses.

Option values are preserved in the CICS IA control file, CIUCNTL, so that they can be used across separate runs of the Collector. For more information about the control file, see “The control record VSAM file” on page 22.

Running the CINT transaction at one CICS terminal, you can:

- Set region-specific options for a single Collector running on the local, or on a remote, CICS region; see “Specifying region-specific options: region configuration.”
- Set region-specific options for each of multiple Collectors running on different CICS regions; see “Specifying region-specific options: region configuration.” You can set options for all the monitored regions simultaneously, provided that the options are to be set to identical values on each region.
- Set global options that apply to all the Collectors you are running; see “Changing global options” on page 114.

Changes take effect only when the Collector is either restarted, Operations action 6 (**Refresh Run Options**) on screen CIU100 is selected, or the CINT REFRESHOPTIONS transaction is issued.

Specifying region-specific options: region configuration

Select 2 Configure Region Options from the Collector Main Administration Menu panel, CIU000 . The Collector Region Configuration Menu panel, CIU200 appears.

SYSID in the New Applid and New Sysid fields, respectively. Type 2 in the Act field and press Enter. CICS IA adds the new region to its list of regions to be monitored, copying the values to all region-specific Collector options from those of the existing region.

3=Delete Region

On a line describing an existing region, type 3 in the Act field and press Enter. CICS IA deletes the region from its list of regions to be monitored.

4=Options

Displays the Resource Options panel, CIU290, which enables you to select, in further panels, what commands are to be monitored.

2 This item allows you to set or change the default values for your chosen operation. You cannot select DEFAULTS with action codes 2 or 3.

3 If more than one CICS region is listed on the Region Configuration Menu, an extra item appears at the top of the list, with an APPLID and SYSID of **ALL**. This item allows you to specify that the chosen operation is to be applied to all the regions in the list. You cannot select **ALL** with action codes 2 or 3.

4 Each line that contains data represents a CICS region. To perform an action against one of the listed regions, type an action code in its Act field and press Enter.

Specifying Resource options: region configuration

The Resource Options panel allows you to specify the resource options to modify. To call up this panel, from the general Region Configuration Menu panel, CIU200, type 4 and press Enter.

The Resource Options panel, CIU290, appears.

```
CIU290          CICS Interdependency Analyzer for z/OS - V3R1M0      2008/03/11
                Resource Options for                               17:36:44
                CICS Sysid: MANA  CICS Applid: IYDZZ320

Type action code then press ENTER.  _  1

  1 = General Options
  2 = Time/Date Options

Interdependency Options          Affinity Options
  3 = CICS Options for APIs      7 = Affinity Options
  4 = CICS Options for SPIs
  5 = DB2/IMS/MQ/TRUE Options
  6 = Natural Options

CICS Sysid: MANA  CICS Applid: IYDZZ320  TermID: TC62

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

Figure 23. Collector Resource Options panel, CIU290

Notes:

1 The action codes that you can specify:

1=General Options

This option allows you to set or modify some region-specific options (for example, which CICS transactions are to be monitored, and whether dependency or affinity data is to be collected) for one or more regions. See “Specifying region-specific options: general.”

2=Time/Date Options

This option calls up the Time and Date Options panel, CIU280, shown in “Specifying region-specific options: timers” on page 103.

On this panel, you can specify whether or not the timer is to be active during this time period. Specify Y (Yes) or N (No), or to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is Y (Yes), but you might have modified it since.

3=CICS Options for APIs

This option allows you to set or modify which dependency-related CICS API commands are to be monitored on one or more regions. See “Specifying region-specific options: API and SPI commands to be monitored” on page 98.

4=CICS Options for SPIs

This option allows you to set or modify which dependency-related CICS SPI commands are to be monitored on one or more regions. See “Specifying region-specific options: API and SPI commands to be monitored” on page 98.

5=DB2/IMS/MQ/TRUE Options

This option allows you to set or modify which dependency-related DB2, IMS, MQ commands, and task related user exits are to be monitored on one or more regions. See “Specifying which dependency-related DB2, IMS, MQ commands and TRUES are to be monitored” on page 100.

6=Natural Options

This option calls up the Natural Options panel, CIU29N, shown in Figure 34 on page 113.

This panel allows the user to configure the collection of data on Adabas and Natural program calls.

7=Affinity Options

This option allows you to set or modify which affinity-related commands are to be monitored on one or more regions. See “Specifying which affinity-related CICS commands are to be monitored” on page 102.

Specifying region-specific options: general

The Region Configuration Menu panel is used for specifying general region-specific options.

You can use the Region Configuration Menu in these ways:

- To specify CICS IA general region-specific options for a particular CICS region, type action code 4 against the APPLID of the region on the Region Configuration Menu panel, CIU200 (shown in Figure 22 on page 91). Press Enter. Type action code 1 on panel CIU290 and press Enter.
- If more than one CICS region is listed on the Region Configuration Menu, to apply your choices to all the regions type action code 4 against **ALL** at the top of the list on panel CIU200 and press Enter. Type action code 1 on panel CIU290 and press Enter.

- To set or change the default values for the general region-specific options, type action code 4 against **DEFAULTS** on the CIU200 panel and press Enter. Type action code 1 on panel CIU290 and press Enter.

The General Options panel, CIU260, shown in Figure 24, is displayed.

Note:

1. If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the General Options panel are set to those of the first “real” CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special **ALL** and **DEFAULT** applids. Both the CICS APPLID and the SYSID are shown as **ALL**.
2. If you specified **DEFAULTS** on the Region Configuration Menu, the initial values of the fields on the General Options panel are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as **DEFAULTS** and the SYSID as **DFTS**. Any changes that you make are saved in the control file and become the new default values.

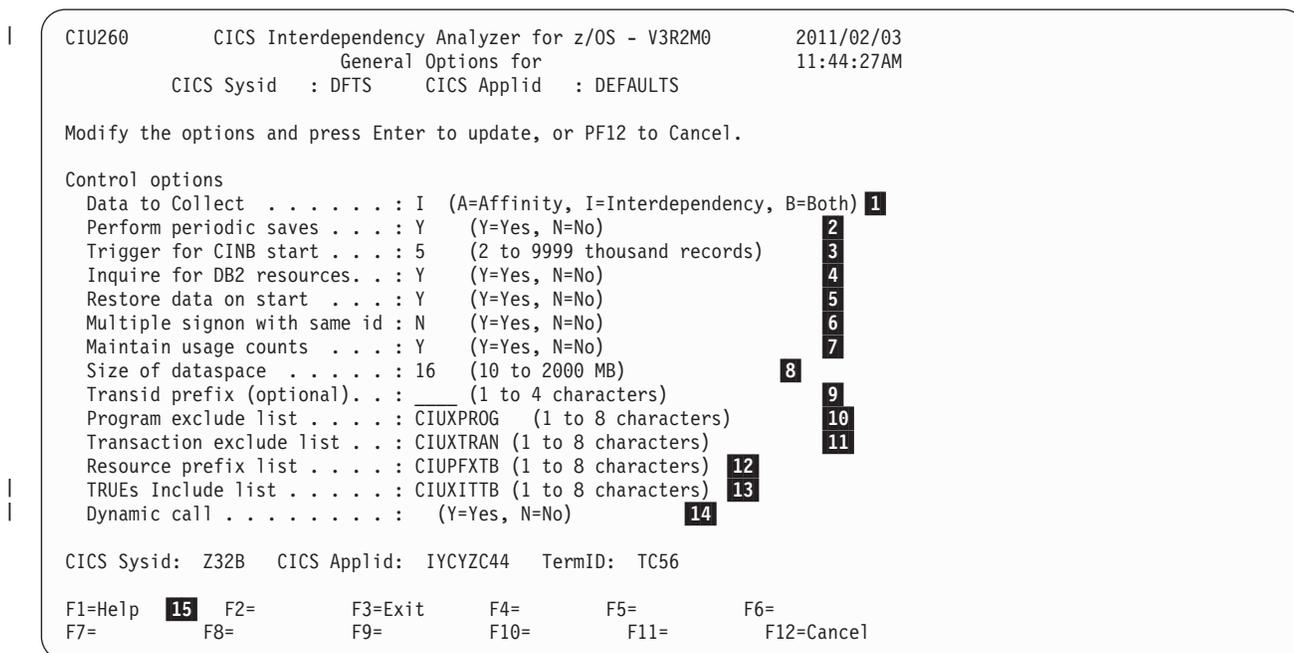


Figure 24. Collector General Options panel, CIU260

Notes:

1 Data to Collect

This field is used to specify which type of data you want to collect: affinity data, interdependency data, or both. Use A for affinity, I for interdependency, or B for both affinity and interdependency data. Alternatively, you can use the default value in the CICS IA control file by leaving the field blank. When the control file is first created, the system default value is I for interdependency, but you might have modified it since.

- You can choose to collect interdependency data on one region and affinity data on another.

- You can change the type of data (interdependency or affinity) to be collected on a region while the Collector is running on that region, but changes do not take effect until the Collector is restarted.

Note: The options specified for the Time and Date Options fields will override the value specified for the **Data to Collect** field when they are set to a value other than Y.

2 Perform periodic saves

This field is used to specify whether you want the collected interdependency data, affinity data, or both to be saved to the VSAM data files, if one of these conditions applies:

- More than 300 seconds have passed since the last save.
- More than the number of table elements specified by the **Trigger for CINB start** option have changed since the last save.
- You pause the Collector. (The autosave transaction, CINB, writes the collected data to the data VSAM data files automatically when you stop the Collector.)

You can specify Y or N, or use the default value in the CICS IA control file to specify a blank. When the control file is first created, the system default value is Y (Yes), but you might have modified it since.

3 Trigger for CINB start

Enter the number of records to be updated to trigger a CINB save, in thousands. You can enter a value from 1 - 9999. A value of 1 indicates that no saves will be triggered. The default is 5.

4 Inquire for DB2 resources

Enter a value of Y for YES and N for NO. If set to YES, the SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT tables are queried to obtain further information. The default is Y.

5 Restore data on start

This field is used to specify whether you want data to be restored from the VSAM data files when the Collector is started, which enables newly collected data to be added to the data collected from previous runs of the Collector. If you are gathering data, use one of the following ways:

- For one set of transaction identifiers at a time
- For one set of commands at a time
- For either interdependency data, affinity data, or both
- If the Collector is being run, at varying times

Setting the option is also of particular value if the Collector stops unexpectedly, because you do not have to start collecting data all over again; you can start from the last time data was saved.

You can specify Y or N, or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is Y (Yes), but you might have modified it since.

6 Multiple sign-on with same ID

If your conventions allow more than one user to be signed on to CICS with the same user ID at the same time, set the Multiple sign-on with same ID field to Y. If you do not, the Collector might incorrectly deduce some affinity lifetimes and create erroneous affinity transaction groups. You also set the field to Y for conventions where more than one user is simultaneously not signed on; that is, they all take the default user ID CICSUSER.

Also, if you are running the Collector in an AOR, the user IDs examined depend on whether the user ID is propagated from the TOR or derived from the SESSION and CONNECTION resource definitions. In the latter case, set the multiple sign-on option to Y if your conventions allow the same AOR user IDs to be signed on to CICS at the same time.

It is very important that this option is set correctly. If you are about to start a new run of the Collector, and intend to restore data from the affinity data VSAM files, ensure that this option is the same as that used in the previous run of the Collector, for which affinity data is to be restored.

You can specify Y or N or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is N (No), but you might have modified it since.

7 Maintain usage counts

This field is used to specify whether the Collector will record the number of times an interdependency or affinity is detected. The usage count tells you how often resources are used by different applications.

If usage data is not required, switch it off. When usage counts are not maintained, the Collector does not have to update an interdependency or affinity record each time it detects an interdependency or affinity that it has already seen. Hence, when the Collector has been running for some time, and it is detecting fewer and fewer new interdependencies or affinities, fewer and fewer records will be saved to VSAM. Thus, you have a useful test for the completeness of the set of detected interdependencies or affinities.

You can specify Y or N, or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is Y (Yes), but you might have modified it since.

8 Size of data space

The size, in megabytes, that you specify for the data space to store the collected data. The size of the data space is fixed for a run of the Collector.

For information about calculating the size of the data space, refer to Appendix F, "CICS IA space considerations," on page 327.

If the data space becomes full while the Collector is running, the Collector stops with abend code IUXB. If the Collector was saving data at the time, a delay might occur from the time the data space becomes full until the time the Collector stops.

The default value is 16.

9 Transid prefix

The 1- to 4-character prefix of the CICS transactions for which you want to gather interdependency or affinity data. If you do not specify any characters, data is collected for all transactions. If you specify a valid prefix (for example, ABC), data is collected only for those transactions with identifiers starting with the prefix. If you specify a blank, the default transid prefix in the CICS IA control file is used.

10 Program exclude list

The 1- to 8-character name of a table containing a list of 1- to 8-character program prefixes. Data is not collected for any program with a name beginning with any of the prefixes. If you specify a blank, the default table name in the CICS IA control file is used. The system default name is CIUXPROG, which is the name of an exclude list supplied with I. For information about how to create your own program exclude lists, see “Creating your own program exclude, transaction exclude, and resource prefix lists” on page 55.

11 Transaction exclude list

The 1- to 8-character name of a table containing a list of 1- to 4-character transaction prefixes. Data is not collected for any transaction with the name beginning with any of the prefixes. If you specify a blank, the default table name in the CICS IA control file is used. The system default name is CIUXTRAN, which is the name of an exclude list supplied with CICS IA. For information about how to create your own transaction exclude lists, see “Creating your own program exclude, transaction exclude, and resource prefix lists” on page 55.

12 Resource prefix list

The 1- to 8-character name of a table containing a list of 1- to 32-character resource prefixes. The resource prefix is used to scan a resource name. If a match is found, the remainder of the resource name is replaced by “+”s. For example, if your application program uses a TSQueue or ENQ/DEQ resource name, which consists of a 3-character prefix “DFH” and a 5-character numerical value, and if you do not have an entry in the resource prefix list for “DFH”, CICS IA records an entry for the EXEC CICS command with all the possible TSQueue resource names: DFH00001, DFH00002, DFH00003... DFHnnnnn. This redundancy might lead to many thousand entries for the same EXEC CICS command. If the “DFH” prefix is added to the list, CICS IA reports only one entry for the EXEC CICS command with a resource name of DFH+++++. The system default name CIUPFXTB is the name of a resource prefix list supplied with CICS IA. For information about how to create your own resource prefix lists, see “Creating your own program exclude, transaction exclude, and resource prefix lists” on page 55. See also “ENQ/DEQ” on page 193 and “TS commands” on page 194 to learn about these resources.

13 TRUEs include list

The 1- to 8-character name of a table containing a list of TRUE names and corresponding to these TRUEs product names. Data is collected for any pair of a TRUE and product name from this list. If you specify a blank, the default table name in the CICS IA control file is used. The system default name is CIUXITTB, which is the name of an include list supplied with CICS IA. For information about how to create your own TRUE include list, see “Creating your own TRUE include list” on page 58.

14 Dynamic call

| This field is used to specify whether you want the dynamic calls to be detected.
| Specify Y (Yes) or N (No). The default value is Y. Specifying Y (Yes) might result in
| an increase in the performance cost while collecting data. If you are aware that
| your environment or your application does not perform dynamic program calls,
| specify N (No).

15 Help

Pressing F1 for help does not save any changes made; you must press the Enter key to save changes.

Specifying region-specific options: API and SPI commands to be monitored

Use the Region Configuration Menu panel CIU200 to specify the API and SPI commands to be monitored.

These two types of dependency-related CICS commands can be collected:

- To specify which API commands are to be monitored on a particular CICS region, type action code 3 at the top of the Resource Options panel, CIU290, shown in Figure 23 on page 92. Press Enter.

The CICS Resources Options panel, CIU240, shown in Figure 25 on page 99, is displayed.

- To specify which SPI commands are to be monitored on a particular CICS region, type action code 4 at the top of the Resource Options panel, CIU290, shown in Figure 23 on page 92). Press Enter.

The CICS Resources Options panel, CIU245, shown in Figure 26 on page 100, is displayed.

Note:

1. If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the CICS Resources Options panel are set to those of the first “real” CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special **ALL** and **DEFAULT** applids. Both the CICS APPLID and the SYSID are shown as ALL.
2. If you specified **DEFAULTS** on the Region Configuration Menu, the initial values of the fields on the CICS Resources Options panel are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as DEFAULTS and the SYSID as DFTS. Any changes that you make are saved in the control file and become the new default values.

If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

```

CIU240          CICS Interdependency Analyzer for z/OS - V3R2M0          2011/02/25
                  CICS Resources Options for                          03:32:45PM
          CICS Sysid : _____ CICS Applid : _____
Modify the options and press Enter to update, or F12 to Cancel.

Detect command types Y=Yes, N=No or blank=default 1
                  D=Yes+Detail, ( Only for API types marked with * ) 3

APIs 2
*Programs . . . _ *Files . . . . _ *Transactions . _ Task Control . _
Presentation . _ *TS Queues . . _ *TD Queues . . _ Journals . . . _
DTP . . . . . _ Counters . . . _ FEPI . . . . . _ *WEB Services . _
*EXITS. . . . . _ Others . . . . _ EVENT proc . . _ ATOMServices . _
XMLtransform . _ WSAddressing . _

CICS Sysid: _____ CICS Applid: _____ TermID: _____

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

```

Figure 25. Collector CICS Resources Options panel, CIU240

Notes:

1 Detect command types

This option determines whether the Collector is to monitor each of the types of dependency-related API command listed. For each type of API command that you want the Collector to monitor, type Y. For each type of API command that you do not want the Collector to monitor, type N or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value for each command is Y (Yes), but you might have modified it since.

Note: If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

2 APIs

The groups of CICS API commands that can be monitored.

The commands are listed by groups on the panel. The individual API commands that make up each group are listed in Dependency-related CICS API commands detected by the CICS Interdependency Analyzer. The individual EXEC CICS FEPI commands that make up the FEPI API group are listed in “CICS FEPI API commands” on page 190.

Note: When collecting Command Flow data for CICS TS V2.2 or earlier, set the TD Queues option to N so that CICS IA collects the Transient Data queue commands correctly. If the TD Queue flag is not set to N, the results are unpredictable.

3 D=Yes+Detail option

You can use option D only for the API types marked with an asterisk.

```

CIU245          CICS Interdependency Analyzer for z/OS - V3R2M0          2011/02/25
                  CICS Resources Options for                          03:32:45PM
          CICS Sysid : _____ CICS Applid : _____
Modify the options and press Enter to update, or PF12 to Cancel.

Detect command types  Y=Yes, N=No or blank=default      1

SPIs (Create/Inquire/Set/Discard/Perform)      2
  Programs . . . _ Files . . . . _ Transactions . _ Temp Storage . _
  Transient Data _ DB2 . . . . . _ DJAR . . . . . _ BRFacility . . _
  Corbaserver . _ TCPIPService . _ FEPI . . . . . _ Journals . . . _
  Library . . . _ IPCONN . . . . _ EVENT proc . . _ Bundles . . . . _
  ATOMServices . _ CSD . . . . . _ XMLTransform . _ MQCONN . . . . _
  JVMServer . . . _

CICS Sysid: _____ CICS Applid: _____ TermID: _____

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

```

Figure 26. Collector CICS Resources Options panel, CIU245

Notes:

1 Detect command types

This option determines whether the Collector is to monitor each of the types of dependency-related SPI command listed. For each type of SPI command that you want the Collector to monitor, type Y. For each type of SPI command that you do not want the Collector to monitor, type N or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value for each command is Y (Yes), but you might have modified it since.

Note: If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

2 SPIs

The groups of CICS SPI commands that can be monitored.

The individual SPI commands that make up each group are listed in “CICS SPI commands” on page 186. The individual EXEC CICS FEPI commands that make up the FEPI SPI group are listed in Dependency-related CICS FEPI SPI commands detected by the CICS Interdependency Analyzer.

For more information about restrictions affecting the monitoring of commands that might cause dependencies or affinities, see “What is not monitored” on page 17 and Appendix A, “Details of dependencies and affinities collected,” on page 179.

Specifying which dependency-related DB2, IMS, MQ commands and TRUEs are to be monitored

Use the Resource Options panel CIU290, to specify which dependency related commands are monitored.

To specify whether dependency-related DB2, IMS, MQ, and RMI True commands are to be monitored on a particular CICS region, type action code 5 at the top of the Resource Options panel, CIU290 (shown in Figure 23 on page 92), and press Enter.

The DB2/MQ/IMS/RMI True Resources Options panel, CIU250, shown in Figure 27, appears.

Note:

1. If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the DB2/MQ/IMS/RMI True Resources Options panel are set to those of the first “real” CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special ALL and DEFAULT applids. Both the CICS APPLID and the SYSID are shown as ALL .
2. If you specified DEFAULTS on the Region Configuration Menu, the initial values of the fields on the DB2/MQ/IMS/RMI True Resources Options panel are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as DEFAULTS and the SYSID as DFTS. Any changes that you make are saved in the control file and become the new default values.

If you are editing the DEFAULTS record, you must specify **Y** or **N** in each field. You cannot specify a blank.

```

CIU250          CICS Interdependency Analyzer for z/OS - V3R2M0          2011/01/26
                DB2/MQ/IMS/RMI True Resource Options for                12:19:28PM

                CICS Sysid   : TLS4   CICS Applid   : IYCLZC04

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

Detect command types (Y=Yes, N=No or blank=default)  1
Fields must be Y or N in the DEFAULTS record

DB2 Options          MQ Options          IMS Options
DB2 . . . . . _     MQ . . . . . _     IMS . . . . . _

RMI True Options
RMI True. . . . . _

CICS Sysid: TLS3   CICS Applid: IYCLZC03   TermID: TC20

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

```

Figure 27. Collector DB2/MQ/IMS/RMI True Resource Options panel, CIU250

Note:

1 Detect command types

This option is used to specify whether or not the Collector is to monitor each of the types of commands listed. For each type of command that you want the Collector to monitor, type **Y**. For each type of command that you do not want the Collector to monitor, type **N**, or, to use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is **N** (No), but you might have modified it since.

The individual commands that make up each of the DB2, MQ, and IMS groups are listed in “Non-CICS API commands detected” on page 191. You cannot select a subset of commands within a group.

If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

For more information about restrictions affecting the monitoring of commands that might cause dependencies, see “What is not monitored” on page 17.

Specifying which affinity-related CICS commands are to be monitored

Use the Resource Options panel CIU290 to specify which affinity related commands are to be monitored.

To specify which affinity-related CICS commands are to be monitored on a particular CICS region, type action code 8 at the top of the Resource Options panel, CIU290, shown in Figure 23 on page 92, and press Enter.

The CICS Affinities Options panel, CIU270, shown in Figure 28 on page 103, is displayed.

Note:

1. If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the CICS Affinities Options panel are set to those of the first “real” CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special ALL and DEFAULT applids. Both the CICS APPLID and the SYSID are shown as ALL .
2. If you specified **DEFAULTS** on the Region Configuration Menu, the initial values of the fields on the CICS Affinities Options panel are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as DEFAULTS and the SYSID as DFTS. Any changes that you make are saved in the control file and become the new default values.

If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

```

CIU270  CICS Interdependency Analyzer for z/OS - V3R2M0      2011/01/17
        CICS Affinities Options for                          01:37:53PM
        CICS Sysid   : Z325   CICS Applid   : IYDZZ325

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

Detect affinity types: Y=Yes, N=No or blank=default

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

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

CICS Sysid: Z325   CICS Applid: IYDZZ325   TermID: TC63

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

```

Figure 28. Collector Affinities Options panel, CIU270

Notes:

1 Detect affinity types

This option is used to specify whether or not the Collector is to monitor each of the types of affinity-related command listed. For each type of command that you want the Collector to monitor, type Y. For each type of command that you do not want the Collector to monitor, type N, or use the default value in the CICS IA control file, specify a blank. When the control file is first created, the system default value is Y **Yes**, but you might have modified it since. If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

2 Inter-transaction

This group of CICS commands can cause inter-transaction affinities.

3 Transaction-system

This group of CICS commands can cause inter-system affinities.

Specifying region-specific options: timers

A timer controls the times and dates at which dependency data, affinity data, or both are collected in a CICS region.

After the Collector is started, the timer automatically pauses and resumes data collection at specified times. You can override the actions of the timer: see “Pausing the collection of data” on page 85 and “Resuming the collection of data” on page 86. However, the timer continues to operate and to pause and resume the collection of data at the specified times.

The timer does not pause or resume data collection exactly on the hour when data is being collected on multiple regions, to minimize contention for shared resources.

Do one of the following:

- To specify CICS IA the Time and Date options for a particular CICS region, type action code 4 against the APPLID of the region on the Region Configuration Menu panel, CIU200, shown in Figure 22 on page 91. Press Enter. Then type action code 2 on panel CIU290 and press Enter.
- If more than one CICS region is listed on the Region Configuration Menu, to apply your choices to all the regions type action code 4 against **ALL** at the top of the list on panel CIU200 and press Enter. Type action code 2 on panel CIU290 and press Enter.
- To set or change the default values for the Time and Date options, type action code 4 against **DEFAULTS** on the CIU200 panel and press Enter. Then type action code 2 on panel CIU290 and press Enter.

The Time and Date Options screen, CIU280, shown in Figure 29, is displayed.

- If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the Time and Date Options panel are set to those of the first “real” CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special **ALL** and **DEFAULTS** applids. Both the CICS APPLID and the SYSID are shown as **ALL**.
- If you specified **DEFAULTS** on the Region Configuration Menu, the initial values of the fields on the Time and Date Options panel are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as **DEFAULTS** and the SYSID as **DFTS**. Any changes that you make are saved in the control file and become the new default values.

If you are editing the **DEFAULTS** record, you cannot specify a blank for any of the slots.

```
CIU280  CICS Interdependency Analyzer for z/OS - V3R1M0          2011/02/09
        Time and Date Options for                               12:21:38PM
        CICS Sysid   : TLS4   CICS Applid   : IYCLZC04

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

Hour slots: Y=Yes, N=No, A=Affinity, I=Interdependency, B=Both, or BLANK
Month, Day, and Week slots: Y=Yes, N=No or BLANK

Hour of day: 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24  1
-----
Day of week: Mon Tue Wed Thu Fri Sat Sun  2
-----
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  3
-----
Month of year: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  4
-----
CICS Sysid: TLS4   CICS Applid: IYCLZC04   TermID: TC23

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

Figure 29. Collector Time and Dates Options screen, CIU280

Notes:

The Hour slots and Month, Day, and Week slots are used to specify whether or not the timer is to be active during this time period and what type of data to

collect. When the control file is first created, the system default value for all of the slots is Y (Yes), but you might have modified it since.

1 Hour of day

Specify the type of data to collect or specify a blank, to use the default value in the CICS IA control file. The types of data values are as follows:

- A, collect Affinity data
- B, collect both Affinity and Interdependency data
- I, collect Interdependency data
- N, do not collect data
- Y, collect data as specified by the **Data to Collect** value on the General Options panel

2 Day of week

Specify Y (Yes) or N (No), or specify a blank to use the default value in the CICS IA control file.

3 Day of month

Specify Y (Yes) or N (No), or specify a blank to use the default value in the CICS IA control file.

4 Month of year

Specify Y (Yes) or N (No), or specify a blank to use the default value in the CICS IA control file.

Note: If you are editing the **DEFAULTS** record, you cannot specify a blank for any of the slots.

Example:

You want the Collector to run every Monday between 1 and 2 a.m. for the next year collecting affinity data. Set 1-2 to A and all the other Hour of day slots to N. Set Mon to Y and all of the other Day of week slots to N. Set all the Day of month and Month of year slots to Y.

Managing Command Flow data collection

You can manage the Command Flow data collection with the aid of the User Command Flow Utility, which helps you to create new configurations of the Command Flow data collection options, change or define parameters of the configurations that already exist, and display statistics about data collection.

Displaying the CICS IA Command Flow Options panel

With the CICS IA Command Flow Options panel you can control the Command Flow data collection.

About this task

You can use the CICS IA Command Flow Options panel to perform the following tasks:

- Start and stop the collection of the Command Flow data in the selected region or regions.

- Submit the batch jobs for loading some predefined data sets (GDG, CSV) and DB2 tables, or flushing data sets (LogStream). **(This option is reserved for future use.)**
- Configure the initial part of the Command Flow data collection options.
- Show information about the current state of the CINC collector.
- Show the start (and stop) date and time of the last data collection and last Command Flow data loading into GDG. **(This option is reserved for future use.)**
- Display the CICS IA Command Flow ApplID list panel, CIUA02, which helps you to configure the second part of the Command Flow data collection options.
- Display the CICS IA User Command Flow Statistics panel, CIUA03, which helps you to view the Command Flow data statistics for all the regions from the user's regions configuration.

Procedure

1. At a CICS terminal, type the transaction identifier CINC.
2. Press Enter. Panel CIUA01 is displayed; see Figure 30.

```

CIUA01          CICS IA Command Flow Options          1 ApplID  IYDZZ41A
Command Flow state . . . . . : STOPPED 2
Date/Time of last start. . . . : 2011/03/25 09:16:18PM 3
Date/Time of last stop . . . . : 2011/03/25 09:17:36PM 4

Command Flow Id . . . . . : TESTUSRI 5

UserID . . . . . : CICSUSER 6
TermID . . . . . : * 7

Transaction list . . . . . : TWEB TSTC 8
                               (Up to 5 transaction IDs)
User Journal Name. . . . . : CIUMTJNL 9

Journal Copy Criteria . . . . : LAST (LAST, USER or CFID) 10

User Exit Name. . . . . : _____ 11
Dynamic Call . . . . . : Y (Yes/No) 12

CIU7000I 5655-U86 (C) Copyright IBM Corp. 2001,2011
F1=Help    F2=      F3=Exit  F4=Options  F5=Start    F6=Stop 13
F7=Stats   F8=      F9=      F10=      F11=      F12=Cancel

```

Figure 30. Command Flow Options panel, CIUA01

The meaning of each part of the CICS IA Command Flow Options panel, CIUA01, is as follows:

1 The 8-character APPLID of the CICS region on which the CICS IA Command Flow Utility is running.

2 The current state of the Command Flow data collection process on all the regions for the user's regions configuration (RUNNING or STOPPED).

RUNNING

The Command Flow Collector is running on all the regions.

PARTLY STA

The Command Flow Collector is running on some of the regions.

|
| **STOPPED**

| The Command Flow Collector is stopped on all the regions

| **START FAIL**

| An error was detected during the Command Flow Collector start process.

| **STOP FAIL**

| An error was detected during the Command Flow Collector stop process.

| **INCOMPLETE**

| The Command Flow start (or stop) process was not completed. Press F6
| (stop) to reset this status.

| **NO APPLIDS**

| The Command Flow APPLID list is empty.

| **3** The date and time when the Command Flow data collection process was last
| started. The time shown is the local time, and the date is given in the format
| specified by the Global Options Menu panel of the CINT transaction.

| **4** The date and time when the Command Flow data collection process was last
| stopped. The time shown is the local time, and the date is given in the format
| specified by the Global Options Menu panel of the CINT transaction.

| **5** The name of the Command Flow trace that is to be captured.

| **6** The user identifier associated with CICS transactions, for which you want to
| collect the Command Flow data.

| **7** The terminal identifier associated with CICS transactions, for which you want
| to collect the Command Flow data.

| **8** The list of transaction identifiers, for which you want to collect the Command
| Flow data. You can enter up to five transaction IDs and you can use wildcard
| characters.

| **9** The journal name for trace data. The length of the name can be up to 8
| characters. The default journal name is CIUMTJNL.

| **10** The LAST, USER or CFID criteria:

- | • LAST specifies that the CIUJLCPY job must copy Command Flow records
| collected at the last Command Flow run.
- | • USER specifies that the CIUJLCPY job must copy all the Command Flow records
| for the USER.
- | • CFID specifies that the CIUJLCPY job must copy the Command Flow records
| that were collected for a specified Command Flow ID.

| **11** The 8-character user modifiable exit name that you can use to add data to a
| user Command Flow records.

| **12** Capturing the dynamic calls. Specify Y (Yes) or not N (No). The default value
| of this field is Y.

| **13** The control keys that you can use to select functions that control the
| operation of the CICS IA User Command Flow Utility and provide help
| information about it. Actions are summarized in Table 17 on page 108.

Table 17. The User Command Flow Utility control keys on the Command Flow Options panel

Action	Function key
Getting help information about the CICS IA User Command Flow Utility.	F1
Starting the Command Flow data collection. The data collecting is started for all the CICS regions from the user's regions configuration of the CICS IA User Command Flow Utility. All the changed user command flow options are saved.	F5
Stopping collecting Command Flow data on all the CICS regions from the user regions configuration.	F6
Displaying the statistics of the Command Flow data collection on all the CICS regions from the user's regions configuration. (The CICS IA User Command Flow Statistics panel, CIUA03, is displayed.)	F7
Accessing to the second part of the Command Flow options in order to view or change it. (The CICS IA Command Flow ApplID list panel, CIUA02, is displayed.) This function does not preserve the Command Flow options changes made on the CIUA01 panel.	F4
Saving all the changed Command Flow options and returning to CICS. Note: The changes of the Command Flow options are rejected if the Command Flow Collector is running.	F3
Saving all the changed Command Flow options without returning to CICS. Note: The changes of the Command Flow options are rejected if the Command Flow Collector is running.	Enter
Returning to CICS without saving any changed Command Flow options.	F12

Notes:

- a. Multiple users on the same CICS region can use the CICS IA Command Flow Utility concurrently. The current state of the CINC collector represents personal working status of the Command Flow data collection process for every user associated with the CINC transaction. The information about the current state includes working status itself (running or stopped) and the date and time when the Command Flow data collection process was last started by a user. **(This option is reserved for future use.)**
- b. You can start the collection of the Command Flow data only when the current state of the CINC collector is stopped for you. The type of the data collected depends on what you have specified for the Command Flow options on the CICS IA Command Flow Options and CICS IA Command Flow ApplID list panels.
3. Press F3 (or F12) to close the CICS IA Command Flow Options panel. Closing the panel does not affect the state of the CINC collector.

Displaying the CICS IA Command Flow ApplID list panel

With the CICS IA Command Flow ApplID list panel you can modify the advanced command flow data collection options.

About this task

You can use the CICS IA Command Flow ApplID list panel to configure the second part of the Command Flow data collection options.

Procedure

1. Press F4 on the CICS IA Command Flow Options panel, CIUA01. The CICS IA Command Flow ApplID list panel, CIUA02, is displayed. See Figure 31.

CIUA02 CICS IA Command Flow ApplID list **1** AppID IYDZZ41A

CICS AppIDs. : IYDZZ41A _____ **2**
 : _____
 : _____

F1=Help F2= F3=Exit F4= Prompt for Regions F5= F6= **3**
 F7= F8= F9= F10= F11= F12=Cancel

Figure 31. Command Flow ApplID list panel, CIUA02

The meaning of each part of the CICS IA Command Flow ApplID list panel, CIUA02, is as follows:

- 1** The 8-character APPLID of the CICS region on which the CICS IA Command Flow Utility is running.
- 2** The list of CICS APPLIDs, for which you want to collect the Command Flow data. User can enter up to 15 APPLIDs.
- 3** The control keys that you can use to select functions that control the operation of the CICS IA User Command Flow Utility or provide help information about it. Actions are summarized in Table 18.

Table 18. The User Command Flow Utility control keys on the Command Flow ApplID list panel

Action	Function key
Getting help information about the CICS IA User Command Flow Utility.	F1
Saving all the changed Command Flow options and returning to the CICS IA Command Flow Options panel, CIUA01. Note: The changes of the Command Flow APPLID list are rejected if the Command Flow Collector is running.	F3
Saving all the changed Command Flow options without returning to the CICS IA Command Flow Options panel, CIUA01. Note: The changes of the Command Flow APPLID list are rejected if the Command Flow Collector is running.	Enter

The meaning of each part of the CICS IA User Command Flow Statistics panel, CIUA03 is as follows:

1 The 8-character APPLID of the CICS region on which the CICS IA Command Flow Utility is running.

2 Table of Command Flow statistics:

- Details in the CICS ApplID column represent the user regions configuration.
- Details in the Record written represent the total number of Command Flow records written to a user journal for each region during the most recent period of data collection.
- The status of the Command Flow collector in the region.

3 The control keys that you can use to select functions that control the operation of the CICS IA User Command Flow Utility or provide help information about it. Actions are summarized in Table 19.

Table 19. The User Command Flow Utility control keys on the User Command Flow Statistics panel

Action	Function key
Getting help information about the CICS IA User Command Flow Utility.	F1
Returning to the CICS IA Command Flow Options panel, CIUA01.	F3 or F12
Redisplaying the CICS IA User Command Flow Statistics panel, CIUA03	Enter

2. Press F3 (or F12) to close the CICS IA User Command Flow Statistics panel and return to the CICS IA Command Flow Options, CIUA01. Closing the panel does not affect the state of the CINC collector.

Displaying the list of available CICS regions

You can use the list of available CICS regions to select the regions that you want to monitor.

About this task

You can use this panel to select the APPLID from the list of available regions for your Command Flow configuration. The selected APPLID is transferred to the CICS IA Command Flow ApplID list panel, CIUA02. You can iteratively select up to 15 APPLIDs.

Procedure

1. Press F4 (Prompt for Regions) on the CICS IA Command Flow ApplID list panel, CIUA02. The list of available CICS regions, CIUA04, is displayed. See Figure 33 on page 112.

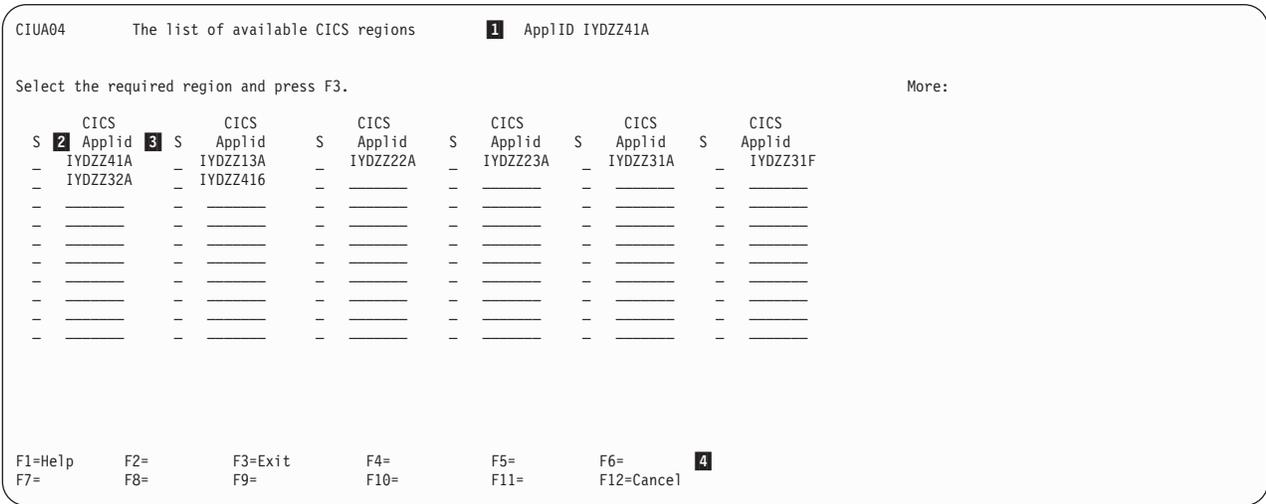


Figure 33. List of available CICS regions , CIUA04

The meaning of each part of the list of available CICS regions menu is as follows:

1 The 8-character APPLID of the CICS region on which the CICS IA Command Flow Utility is running.

2 The region selection fields column (S). It is a 1-character field. You can select any region from the CICS Applid column for your regions configuration by entering any non-blank character in this selection field.

3 The CICS APPLID fields column. All the regions from this list are connected to your CICS region.

4 The control keys that you can use to select functions that control the operation of the CICS IA User Command Flow Utility or provide help information about it. Actions are summarized in Table 20.

Table 20. The User Command Flow Utility control keys on the CIUA04 panel

Action	Function key
Getting help information about the CICS IA User Command Flow Utility.	F1
Getting all the selected CICS APPLIDs and returning to the CICS IA Command Flow ApplID list panel, CIUA02.	F3
Returning to the CICS IA Command Flow ApplID list panel, CIUA02, without getting any selected CICS APPLIDs.	F12
Displaying the CIUA04 panel again.	Enter

Note: The list of available CICS regions panel displays up to 60 regions.

2. Press F3 (or F12) to close the list of available CICS regions and return to the CICS IA Command Flow ApplID list panel, CIUA02. Closing the panel does not affect the state of the CINC collector.

Specifying Natural options

The Natural Options screen allows you to configure the collection of data on Adabas and Natural program calls.

Note:

1. If you specified "ALL" on the Region Configuration Menu, the initial values of the fields on the Natural Options screen are set to those of the first "real" CICS region listed on the Region Configuration Menu; that is, the values are taken from the first region listed after the special "ALL" and "DEFAULT" applids. Both the CICS APPLID and the SYSID are shown as "ALL" .
2. If you specified "DEFAULTS" on the Region Configuration Menu, the initial values of the fields on the Natural Options screen are the defaults, taken from the CICS IA control file, CIUCNTL. The CICS APPLID is shown as "DEFAULTS" and the SYSID as "DFTS". Any changes that you make are saved in the control file and become the new default values.
If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

To call up this screen, go to the Resource Options screen, CIU290, type action code 7, and press Enter. The Natural Options screen, CIU29N, looks as follows:

```
CIU29N          CICS Interdependency Analyzer for z/OS - V3R2M0          2011/01/25
                Natural Resource Options for                          10:23:32AM
                CICS Sysid: DFTS CICS Applid: DEFAULTS

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

Detect command types: Y=Yes, N=No

Adabas Calls . . . . . Y 1          Program Calls . . . . . Y 2

CICS Sysid: Z328 CICS Applid: IYDZZ328 TermID: TC33

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

Figure 34. Collector Natural Options screen, CIU29N

Notes:

1 ADABAS Calls

Enter Y (Yes) to capture ADABAS calls within your Natural CICS environment.

Note: If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

2 Program Calls

Enter Y (Yes) to capture Natural program calls within your Natural CICS environment.

Note: If you are editing the DEFAULTS record, you must specify Y or N in each field. You cannot specify a blank.

Changing global options

Use the Global Options Menu panel to modify the Collector global options.

Select 3 Configure Global Options from the Collector Main Administration Menu panel, CIU000 (shown in Figure 15 on page 72). The Collector Global Options Menu panel, Figure 35, is displayed. From this panel, you can set or modify the following global options that apply to all the regions being monitored:

- Whether VSAM file sharing is to be used
- Whether high-level tracing is on or off
- Whether CICS TS tracing is on or off
- The National Language to be used in screens and messages
- The date and time formats to be used in screens and messages
- The dump HLQ to be produced

```
CIU300          CICS Interdependency Analyzer for z/OS - V3R1M0          2011/01/19
                  Global Options Menu                                12:20:28PM

Modify the options and press Enter to update, or press F12 to cancel.

Control options
VSAM file sharing . . . . . : Y (Yes/No) 1
High Level Trace . . . . . : N (1/2/3/No) 2
Restore Trace flags. . . . : N (Yes/No) 3

National Language Option . . . : E Code: ENU 4

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

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

Dump HLQ . . . . . : 6

CICS Sysid: TLS3 CICS Applid: IYCLZC03 TermID: TC20

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

Figure 35. Collector Global Options Menu panel, CIU300

Notes:

1 VSAM file sharing

This option is used to specify whether the Collector is to save the dependency and affinity data for all monitored CICS regions to a single set of VSAM files that you have shared between regions using either VSAM RLS or function shipping, or to multiple separate, region-specific, sets of files. Type Y or N. The default is N (No), data will be saved in separate sets of files.

To control the operation of multiple instances of the Collector, running on different regions, from a single CICS terminal, set this option to Y.

2 High Level Trace

This option is used to specify whether the high-level trace is to be turned on for the Collector. Type 1, 2, or 3 to specify which level of trace points you want to collect, or type N to turn off this option. The default is N (No), trace is turned off.

The CICS IA trace points are of the following types:

- The points of level 1 are included in CICS IA modules
- The points of level 2 are included in CICS IA modules
- The points of level 3 are included in GLUE or TRUE

The CICS TS trace points are of the following types:

- The points of level 1 are included at the entry and exit of a module
- The points of level 2 are included in all other cases
- The points of level 3 are included in GLUE or TRUE

You can set this levels through the CINT transaction.

3 Restore Trace flags

This option is used to specify whether to restore Trace flags values after a CICS IA stop. Type Y or N. The default is N (No), trace is turned off.

4 National Language Option

The 1-character code for the National Language to be used by CICS IA. The possible values are as follows:

E US English.

K Japanese. For Japanese to be displayed, the terminal must be configured with the CICS SOSI attribute.

5 Date and Time Formats

To set the date format to be used by CICS IA, type the corresponding number (1 - 4) in the Date field. The default is 4. Type the separator character to be used in dates (for example, / or –) in the Date Separator field. The default character is “/”.

To set the time format to be used by CICS IA, type the corresponding number (1 for 12-hour clock format or 2 for 24-hour clock format) in the Time field. The times are shown in the local time. The default is 1. Type the separator character to be used in times in the Time Separator field. The default character is “:”.

6 Dump HLQ

This field is used to define the high-level qualifier of the dump data set that will be produced.

Removing duplicate resources from the Collector files

With the cleanup batch utility, you can remove duplicate resources to minimize the volume of the data stored in the Collector files.

CICS IA provides a batch utility to remove the collected resource types, which have a resource name that consists of a constant prefix and a variable. These types of resource name might lead to a large amount of duplicate data stored for the same command in a particular transaction and program combination.

To avoid collecting unnecessary data, set up a resource prefix table. See “Creating a resource prefix list” on page 57.

The batch utility consists of the CIUJCLDL sample batch job, which calls the CIUDEL program. With this utility, you can clean up collected TSQueue CICS resource prefixes, ENQueue CICS resource prefixes, and the detailed record associated with a TSQueue. You can remove data from all of the CICS regions or from one chosen region.

Running the CICS IA VSAM cleanup utility

Learn how to run the CICS IA clean up utility.

For information about the dependency-causing program commands that you can clean up see “Dependency-related commands” on page 6.

You can clean up the following program commands:

- The transactions and commands that you have specified to be monitored by the Collector. See “Specifying region-specific options: API and SPI commands to be monitored” on page 98.
- The command types that you specify in the REPTYPE DD statement of the cleanup job. See “Modifying a clean up utility job.”
- The CICS command groups that you specify in the CMDGRPS DD statement. See “Modifying a clean up utility job.”

If your dependency data files are shared by multiple regions, you can run only one cleanup job to remove resources whether from one particular region or from all the regions. However, if each monitored region has its own region-specific set of the dependency data files, you must run the job multiple times against the relevant dependency files for each region to clean up data from multiple regions.

Modifying a clean up utility job

To request a resource cleanup from the dependency VSAM files, edit and run the CIUJCLDL job.

Before you run the CIUJCLDL job, modify the following parameters:

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The STEPLIB DD statement

Specify the name of the CICS IA load library in which you have installed the cleanup utility program, CIUDEL.

The CIUINT1 DD statement

Specify the name of the CICS dependency data file for the chosen region.

Each dependency data file has a header record, which specifies the APPLID of the CICS region that creates the record. The cleanup utility checks the correspondence between the APPLID of the region and the APPLID recorded in the control record file, CIUCNTL, and proceeds only if they match.

If the dependency data files and the control record file are shared by multiple regions, they contain a header record for each of the monitored regions.

The CIUINT2 DD statement

Specify the name of the DB2 dependency data file for the chosen region.

The CIUINT3 DD statement

Specify the name of the MQ dependency data file for the chosen region.

The CIUINT4 DD statement

Specify the name of the IMS dependency data file for the chosen region.

| **The CIUINT5 DD statement**

| For the chosen region, specify the name of the CICS dependency data file in
| which you want to store the dependency resources with the names that are
| longer than 32 bytes.

| **The CIUINT6 DD statement**

| Specify the resource detail data file for the chosen region.

| **The CIUCNTL DD statement**

| Specify the name of your CICS IA VSAM control file for the chosen CICS
| region.

| **The APPLID DD statement**

| Specify the APPLID DD statement as follows:

- | • Enter a region VTAM[®] APPLID, if you want to clean up the dependency
| data from a particular region.
- | • Select **ALL**, if you want to clean up the data from all the regions in the
| dependency data files.

| Leaving the field blank also causes cleanup the data from all the regions.

| **REPTYPE**

| Specify the type of command for which you want to clean up the dependency
| data.

| **Note:** You can specify only CICS commands.

| **The CMDGRPS DD statement**

| Specify the CICS command groups for which you want to clean up the
| dependency data.

| **Note:** This statement is effective only if you have specified CICS commands as
| a REPTYPE.

| You can specify the CICS command groups directly from the commands
| selected to be monitored by the Collector.

| The following command groups are supported:

- | • TS commands (READQ TS, WRITEQ TS, DELETEQ TS)
- | • TC commands (ENQ, DEQ)

| You can enter one of these command groups or both of them.

| **Note:** Make sure that there is no prefixes superposition, when using more than
| one command group.

| **The PRFGRPS DD statement**

| Specify the resource prefixes as follows:

- | • Each prefix must be written in a separate string.
- | • The maximum length of a prefix is 32 symbols.
- | • The first symbol of a prefix must not be blank, because it indicates the end
| of a prefix.
- | • The empty string indicates the end of a file.
- | • The empty file means that all the records for the specified command group
| must be deleted.
- | • The **ALL** prefix in the first and the only string also means that all the
| records for the specified command group must be deleted.

| The number of prefixes is not limited and they may be written in any order.

SYSPRINT DD statement

Specify where to save the report produced by the cleanup utility.

Note: The cleanup utility requires correct access to the CICS IA VSAM files. Make sure that the CICS IA Collector is stopped and the VSAM files are unavailable for batch update processing. If necessary, close the VSAM files with the CEMT SET FILE transaction.

The output report consists of a list of the selected command groups and a report about the deleted records. For an example of the output report, see Figure 36.

Generated by CICS Transaction Inter-dependency Utility on

```
2011/01/20 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320
--LIST OF CICS COMMAND GROUPS--

Command Group Deletion
-----

TS          Yes

2011/01/20 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320
DELETED RECORDS REPORT FOR      APPLID: IYDZZ22B
                                FROM CICS RESOURCE FILE

Command Group    Prefix                Amount
-----
ENQ_CMDG        XXXXXXXXXXXXXXXXX      12
TSQ_CMDG        YYYYYYYY                18
TSQ_CMDG        ZZZZ                    18

2011/01/20 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320
DELETED RECORDS REPORT FOR      APPLID: IYDZZ22B FROM
                                FROM RESOURCE DETAIL FILE

Command Group    Prefix                Amount
-----
TSQ_CMDG        YYYYYYYY                3
TSQ_CMDG        ZZZZ                    3
```

Figure 36. Example clean up utility report

Operating CICS IA Collection from the CICS IA Explorer plug-in

In CICS IA V3.2, you can manage data collection not only with the CINT transaction, but also with the CICS IA Explorer plug-in.

The CINT transaction enables you to configure CICS IA to manage multiple regions. To do this, you need to:

1. Share the CIUCNTL control file. For details, see *Creating VSAM files*.
2. Define the CONNECTION and SESSION resources between the controlling CICS region and the regions you want to manage.

In CICS IA V3.2, you can use the CICS IA Explorer plug-in to operate the dependency and affinity data collection. You can START, STOP, PAUSE, CONTINUE, and REFRESH the collection. You can also operate and administer the Command Flow feature by *userid*.

To be able to administer data collection tasks with the Explorer plug-in, define further resources to the region, from which you want to manage CICS IA. Since

CICS IA uses Atomservices resources to ensure seamless interaction between the components, your controlling region must be at CICS TS 4.1 or above.

Note: This requirement is critical for the controlling region only. For the regions you want to manage previous versions of CICS TS will suffice.

In the configuration EXEC, you need to configure the selected region to include the above mentioned resources.

Explorer Plugin variables:

```
CICS IA CONTROL REGION . . . . . ___ (YES or NO)      1
URI PATH . . . . . /CICSIA/IBM/atom/ciuatom/*         2
TCP IP PORT . . . . . _____                     3
ATOM CONFIG FILE LOCATION . . . _____           4
```

Figure 37. Configuring Explorer plug-in variables

Notes:

- 1** Select YES to change the region status to controlling.
- 2** Specify a URI PATH. The default path name is /CICSIA/IBM/atom/ciuatom/*.
- 3** Specify the TCP/IP port number.
- 4** Specify the Atom Configuration file location. The default value is @pathprefix@/usr/lpp/cicsia/@ciuver@. This will be concatenated with IBM/atomservices/config/CIUATM01 to provide the HFS location of the configuration file used by the CICS IA Atomservice. The @pathprefix@ and @ciuver@ are variables chosen during the SMPE install jobs CIUIHFS0, CIUIHFS1, and CIUMKDIR.

Once you have run the configuration for the controlling region, you need to define the configured resources to CICS. For details, see Defining resources to CICS.

Your controlling region should now be ready to enable CICS IA tasks administration form the Explorer plug-in. For more information on configuring the CICS IA Explorer plug-in, refer to the its built-in HELP.

Collector errors

If the CINT, CINB or CINC transaction or an exit program, encounters a serious error the Collector stops by terminating CINT, CINB and CINC.

CINT, CINB and CINC termination codes:

- A code in the IUxx range accompanied by messages on the CINT transient data queue that indicate the cause of the error.
- One of the other CICS transaction abend codes. For a description of the code, see the *CICS Messages and Codes* manual.

If the CINT transaction stops with code ATCH, ATNI, or AKCT, the Collector continues to collect data. For all other codes from either the CINT, CINB or CINC transaction, the Collector is stopped. After performing the actions suggested by the message explanations, you can restart the Collector.

If a program check or MVS abnormal termination occurs in an exit program, it results in an abnormal termination of the transaction that caused the exit to be invoked, probably indicating a problem with the Collector. Use the CINT transaction to stop the Collector and contact your IBM Support Center if the evidence points to the Collector being at fault.

A termination code of AICA, might be caused by the Collector scanning the table of dependency or affinity data when the amount of data is very large. You can prevent this problem by increasing the value of the ICVR system initialization parameter.

Chapter 6. Updating the Dependency and Affinity database objects

The Dependency database objects contain accumulated data about your applications and the resources that they use. Update the database objects regularly to add new information recorded by the Collector in the VSAM dependency files.

The Affinity database objects contain accumulated data about your applications and the potential affinities that might affect them. Update the database objects regularly to add new information recorded by the CICS IA Collector, in the VSAM affinity files.

Updating the Dependency database objects

CICS IA provides a number of batch jobs to update the database from the VSAM files.

CIURES LD

Updates the CIU_RESOURCE table used by the CICS IA plug-in for CICS 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.

CIUUPDBN

Updates the Natural table, CIU_NATURAL_DATA, from the Natural dependency data file, CIUINT7.

CIUUPDB

Updates *all* tables (CICS, DB2, MQ, Natural, and IMS), from all the dependency data files, CIUINT1 through CIUINT7.

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

If, on the other hand, you have separate sets of dependency data files for each region, you can run the jobs once for each region in which you are interested and feed the dependency information for each region into the same set of Dependency database objects. For example, you must run multiple jobs if you: run the Collector on multiple regions, do not use a shared set of dependency files, and want to keep the dependency data for all the regions in the same set of Dependency database objects.

Run the Collector with your normal weekly workload to collect the base data and also run it at month ends, quarter ends, and year ends to collect data from infrequently run programs. When you have recorded all your applications, you might need to update the database only for new applications and changes.

Database update procedure

The job streams to update the dependency tables are put into the hlq.SCIUSMP2 library by the CICS IA installation procedure, where "hlq" is a prefix defined during installation.

Before running any of them, you must edit the dependency tables to meet the requirements of your system.

Updating the Affinity database objects

CICS IA provides the CIUAFFLD batch job to update the database from the VSAM files.

If your affinity data files are shared by multiple CICS regions, you can use a single run of the CIUAFFLD job to store the affinity information for all the regions into the set of Affinity database objects.

If, on the other hand, you have separate affinity data files for each region, you can run the job once for each region you're interested in and feed the affinity information for each region into the same set of Affinity database objects. For example, you must run multiple jobs if you run the Collector on multiple regions, do not use shared affinity files, and want to keep the affinity data for all the regions in the same set of Affinity database objects.

Run the Collector with your normal weekly workload to collect the base data and also run it at month ends, quarter ends, and year ends to collect data from infrequently run programs. When you have recorded all your applications, you might need to update the database objects only for new applications and changed programs.

The CIUSPAFF program is a DB2 UDB stored procedure that is an affinity data update and query interface program. The CIUAFFL1 program, which is run by the CIUAFFLD JCL sample, issues one SQL CALL to the CIUSPAFF program that updates affinity DB2 objects.

Before running CIUAFFLD, which is put into the hlq.SCIUSMP1 library by the CICS IA installation procedure, edit it to meet the requirements of your system.

Updating the Command Flow database objects

CICS IA provides batch jobs to update the database from the command flow log stream.

CIUJLCPY

Copies command flow data from the log stream to a GDG dataset.

CIUJLDEL

Deletes data from the log stream. This job must be used only for a non-shared single user log stream.

CIUUPDB5

Updates the CIU_CMDFLOW_DATA table from the GDG dataset.

Chapter 7. Creating and updating a CICS IA UDB database

CICS IA provides jobs to unload CICS IA data into CSV files. These files can be used to populate the IBM supported UDB databases.

CICS IA also provides the Definition Data Language (DDL) sample to create a UDB on Windows NT. This DDL is provided only as a sample. CICS IA also supplies sample tasks to run in your UDB environment. These sample tasks help you to load the tables from the CSV files. The process of creating a UDB CICS IA database usually comprises the following steps:

1. Review and edit a sample DDL provided in hlq.SCIUSMP2(CIUUDBC).
2. Run the DDL to create your UDB CICS IA database objects:
 - TABLES
 - VIEWS
 - Stored Procedures

When you have created a UDB IA database, perform the following steps to load the data:

1. Run the supplied sample jobs to unload the data from the VSAM files into CSV files. See "Preparing CSV files."
2. After you have created your CSV files, transfer them in ASCII to your DB2 Windows NT Server workstation.
3. Load the CSV files by using a UDB load utility. At this point, you can choose either to use a standard load utility or to write a custom update program. If you use a UDB IMPORT facility to load the CSV files, do the following:
 - a. Select the INSERT option.
 - b. Select the DELIMITER options as follows:
 - 1) The column delimiter is a comma (,).
 - 2) The character string delimiter is a double quote (").
4. For affinity collection, run the CIUSPAFF stored procedure to load the affinity tables used by the Explorer. A sample task is provided in hlq.SCIUSMP1(CIUUDBT3).
5. After you have loaded any of the dependency tables, reload the CIU_RESOURCE table used by the Explorer. A sample task is provided in hlq.SCIUSMP2(CIUUDBT1).
6. Before connecting to the Explorer, load the CIU_VERSION table. A sample LOAD task is provided in hlq.SCIUSMP2(CIUUDBT2).

Preparing CSV files

Before you can update a non-DB2 database with new data, you need to convert the VSAM files created by the Collector to the QSAM flat files or, in other words, Comma Separated Values (CSV) files.

CICS IA provides the following batch jobs that you can use for managing dependency tables:

CIUADB

Prepares QSAM CSV files for all Dependency tables including DETAILED tables.

| **CIUADB4**

| Prepares QSAM CSV files for the CIU_CMDFLOW_DATA table.

| **CIUADBAF**

| Prepares QSAM CSV files for the CIU_AFF_EVENTS tables.

| **CIUADBAP**

| Prepares QSAM CSV files for the CIU_APPLS_RESOURCE and
| CIU_APPLS_DESC tables.

|
| When creating a new CSV file for the Dependency tables, you can associate the
| collection by using a new column, COLLECTION_ID, in the tables. You can use the
| same COLLECTION_ID for each load or select a new one (recommended in case of
| major changes in your application or environment). The CICS IA Explorer plug-in
| enables you to compare resources across COLLECTION_IDs. You can also manage
| your CICS IA data by COLLECTION_ID: for example, delete it from a table.

Chapter 8. Running the Reporter

The Reporter consists of:

- The Dependency Reporter: described in “Running the Dependency Reporter”
- The Affinities Reporter: described in “Running the Affinities Reporter” on page 133

Running the Dependency Reporter

This section describes how to run the CICS IA Dependency Reporter. The Dependency Reporter consists of a batch job, CIUJCLRP, that produces reports of the dependencies found by the Collector.

The dependency-causing program commands that can be reported on are those listed in “Dependency-related commands” on page 6. The program commands that are actually reported depend on:

1. The transactions and commands that you specified to be monitored by the Collector. See “Specifying region-specific options: API and SPI commands to be monitored” on page 98.
2. The types of command, CICS, DB2, MQ, IMS, or ALL, that you specify to be reported on the REPTYPE DD statement of the Reporter job; see REPTYPE in “Modifying a Dependency Reporter Job”.
3. If CICS commands are to be reported, the CICS command groups that you specify on the CMDGRPS DD statement of the Reporter job; see the CMDGRPS DD statement in “Modifying a Dependency Reporter Job.”

If your dependency data files are shared by multiple regions, you can run one Reporter job to produce a report showing dependencies, for example, dependencies on CICS resources or dependencies on DB2 resources, found in either of the following:

- A single, specified, region
- All of the regions

If, on the other hand, each monitored region has its own, region-specific, set of dependency data files, each Reporter job always retrieves data for a single region; to retrieve data from multiple regions you must run your job multiple times, against the relevant dependency files for each region in which you are interested.

To request a report from the Reporter, edit and run the CIUJCLRP job.

Modifying a Dependency Reporter Job

Changes to make before running the CIUJCLRP job.

The JOB accounting parameters

Modify the JOB card statement to meet your site standards.

The STEPLIB DD statement

Specify the name of the CICS IA load library in which you have installed the Dependency Reporter program, CIUREP.

The CIUINT1 DD statement

Specify the name of the CICS dependency data file for this region.

- Each dependency data file has a header record that specifies the APPLID of the CICS region that created the record. The Dependency Reporter checks this APPLID with the APPLID recorded in the control record file, CIUCNTL, and only proceeds if they match.
- If the dependency data files and the control record file are shared across multiple regions, they contain a header record for each of the monitored regions.

The CIUINT2 DD statement

Specify the name of the DB2 dependency data file for this region.

The CIUINT3 DD statement

Specify the name of the MQ dependency data file for this region.

The CIUINT4 DD statement

Specify the name of the IMS dependency data file for this region.

The CIUINT5 DD statement

Specify the name of the CICS "+32" dependency data file for this region. This CICS dependency file that records dependencies on CICS resources with names longer than 32 bytes.

The CIUCNTL DD statement

Specify the name of your CICS IA control VSAM file for this CICS region.

The APPLID DD statement

Specify this statement as follows:

APPLID

Specify the VTAM application identifier (APPLID) of the CICS region for which you want dependency data to be reported. Dependency data is returned for the specified region only.

ALL Returns data for all regions in the dependency data files.

blank Causes data for all regions in the dependency data files to be returned.

REPTYPE

Specify the type of command, CICS, DB2, MQ, IMS, or all, that you want to be included in the report:

CICS Only CICS commands are to be included.

DB2 Only DB2 commands are to be included.

MQ Only MQ commands are to be included.

IMS Only IMS commands are to be included.

ALL All types of command are to be included.

blank All types of command are to be included.

The CMDGRPS DD statement

Specify the CICS commands and command groups that are to be reported. Enter each one on a separate line.

The statement is effective only if you have specified, on the REPTYPE statement, that CICS (or all) commands are to be included in the report.

The command groups that can be specified map directly to those that can be selected on the Collector. If no command or group is given, all detected interdependencies are reported. If one or more commands or command groups is specified, only they are reported.

Here are the commands and command groups that you can specify, with the commands that are reported in each case:

BMS SEND MAP, RECEIVE MAP

COUNTER

DEFINE COUNTER, DEFINE DCOUNTER, DELETE COUNTER, DELETE DCOUNTER, GET COUNTER, GET DCOUNTER, QUERY COUNTER, QUERY DCOUNTER, REWIND COUNTER, REWIND DCOUNTER, UPDATE COUNTER, UPDATE DCOUNTER

DTP ALLOCATE SYSID, ALLOCATE SESSION, CONNECT PROCESS, SEND CONVID, SEND SESSION, CONVERSE CONVID, CONVERSE SESSION, FREE

FC READ, WRITE, DELETE, STARTBR, READNEXT, READPREV, ENDBR, RESETBR, UNLOCK, REWRITE

FEPI All the FEPI API commands listed in Dependency-related CICS FEPI API commands detected by the CICS Interdependency Analyzer

HANDLE

HANDLE ABEND PROGRAM

IBRFA

INQUIRE BRFACILITY, SET BRFACILITY

ICORB

CREATE CORBASERVER, INQUIRE CORBASERVER, SET CORBASERVER, PERFORM CORBASERVER, DISCARD CORBASERVER

IDB2 CREATE DB2ENTRY, CREATE DB2TRAN, INQUIRE DB2ENTRY, INQUIRE DB2TRAN, SET DB2ENTRY, SET DB2TRAN, DISCARD DB2ENTRY, DISCARD DB2TRAN

IDJAR

CREATE DJAR, INQUIRE DJAR, PERFORM DJAR, DISCARD DJAR, INQUIRE JVMPROFILE

IFEPI All the FEPI SPI commands listed in Dependency-related CICS FEPI SPI commands detected by the CICS Interdependency Analyzer

IJRNL INQUIRE JOURNALNAME, INQUIRE JOURNALNUM, SET JOURNALNAME, SET JOURNALNUM, DISCARD JOURNALNAME, DISCARD JOURNALNUM

ISFC CREATE FILE, INQUIRE FILE, SET FILE, DISCARD FILE

ISPC CREATE PROGRAM, INQUIRE PROGRAM, SET PROGRAM, DISCARD PROGRAM

ISTD CREATE TDQUEUE, INQUIRE TDQUEUE, SET TDQUEUE

ISTR CREATE TRANSACTION, INQUIRE TRANSACTION, SET TRANSACTION, DISCARD TRANSACTION

ISTS CREATE TSMODEL, INQUIRE TSMODEL, DISCARD TSMODEL, INQUIRE TSPool, INQUIRE TSQUEUE, SET TSQUEUE, INQUIRE TSQNAME, SET TSQNAME

ITCP CREATE TCPIPSERVICE, INQUIRE TCPIPSERVICE, SET TCPIPSERVICE, DISCARD TCPIPSERVICE

JRNL WRITE JOURNALNUM, WRITE JOURNALNAME, WAIT JOURNALNUM, WAIT JOURNALNAME

LINK LINK

LOAD LOAD, dynamic COBOL calls

OTHER ADDRESS CWA, ASSIGN APPLID

RETURN RETURN TRANSID

START START

TC ENQ, DEQ

TD READQ TD, WRITEQ TD, DELETEQ TD

TS READQ TS, WRITEQ TS, DELETEQ TS

WEB WEB ENDBROWSE FORMFIELD, WEB ENDBROWSE HTTPHEADER, WEB EXTRACT, WEB READ FORMFIELD, WEB READ HTTPHEADER, WEB READNEXT FORMFIELD, WEB READNEXT HTTPHEADER, WEB RECEIVE, WEB RETRIEVE, WEB SEND, WEB STARTBROWSE FORMFIELD, WEB STARTBROWSE HTTPHEADER, WEB WRITE HTTPHEADER

XCTL XCTL

SYSPRINT DD statement

Specify the destination for the report produced by the Dependency Reporter.

The Reporter cannot read from a dependency data file while the Collector has the file open for update, so you cannot run the Reporter if the Collector is RUNNING or PAUSED.

Output from the Dependency Reporter

The Dependency Reporter produces sample reports. Here are examples of CICS, DB2, MQ, and IMS reports.

CICS report

A CICS report consists of a header page and, if the dependency data file contains interdependency records, one or more pages of interdependency records.

Figure 38 on page 129 shows an example header page and Figure 39 on page 130 an example page of interdependency records.

Command Type	Reporting
-----	-----
START	Yes
XCTL	Yes
LOAD	Yes
LINK	Yes
RETURN	Yes
HANDLE	Yes
TC	Yes
FC	Yes
BMS	Yes
TS	Yes
TD	Yes
JRNL	Yes
DTP	Yes
COUNTER	Yes
FEPI	Yes
WEB	Yes
OTHER	Yes
ISPC	Yes
ISFC	Yes
ISTR	Yes
ISTS	Yes
ISTD	Yes
IDB2	Yes
IDJAR	Yes
IBRFA	Yes
ICORB	Yes
ITCP	Yes
IFEPI	Yes
IJRNL	Yes

1
2

Figure 38. Example CICS report from the Dependency Reporter—header page

Notes:

1 Incorrect command types

On an interdependency report that includes CICS commands (that is, where the REPTYPE statement is specified as “CICS” or “ALL”), this area lists any CICS command groups that were specified incorrectly on the CMDGRPS DD statement.

2 Command types reported

On an interdependency report that includes CICS commands, this list shows all the command types that are to be reported. Command groups selected on the CMDGRPS DD statement are marked Yes, and those not selected No.

Tran	Program	Offset	Command	Resource				
		Sysid	Usage	First Run	Last Run	Term	TCBmode	
D8CS	DSN8CC0	000009EA ----	RECEIVE MAP 16	2011-01-27 09.34.34	2011-01-27 09.35.06	Y	QR	DSN8CCD
		00000B24 ----	RECEIVE MAP 27	2011-01-27 09.34.34	2011-01-27 09.35.29	Y	QR	DSN8CCG
		00000D62 ----	LINK PROGRAM 51	2011-01-27 09.34.34	2011-01-27 09.36.09	Y	QR	DSN8CC1
		00001020 ----	SEND MAP 27	2011-01-27 09.34.34	2011-01-27 09.35.24	Y	QR	DSN8CCG
		0000106A ----	RETURN TRANSID 27	2011-01-27 09.34.34	2011-01-27 09.35.24	Y	QR	D8CS

Figure 39. Example CICS report from the Dependency Reporter—main body

The report lists all the CICS commands that were collected by the Collector and (because this is a CICS dependency report) selected in the CMDGRPS DD statement of the CIUJCLRP job. The report entries contain the following information:

Tran The identifier of the CICS transaction under which the command was issued.

Program The name of the program that issued the command.

Offset The offset, from the load point of the program, of the BALR instruction of the command detected. The Dependency Reporter produces an offset of -1 (X'FFFFFFF') if it could not determine an offset; that is, if either of the following is true:

- The program was defined with RELOAD(YES).
- The offset calculated is not within the program, which might indicate that the program or perhaps language runtime code has passed control to another program by using a non-CICS mechanism, for example, a VS COBOL II dynamic call.

Note:

1. This offset is not the same as the offset given by the Load Module Scanner. For an EXEC CICS command, the offset given by the Load Module Scanner is the offset of the command argument 0 declaration from the start of the load module. See page Appendix B, “Correlating Load Module Scanner and Dependency Reporter output to source,” on page 199.
2. An output of X'FFFFFFF' indicates that individual commands cannot be directly located in a program. The program must be scanned for all instances of the command, because more than one instance might be present.

Command

The command that was detected.

Resource

The resource against which the command was issued, causing a dependency between it and the named program.

Sysid If the command was shipped or routed to a remote CICS region, the system identifier of the remote region.

Usage The number of times that this command, with the same CICS APPLID, SYSID, transaction, program, offset, resource, remote SYSID, remote name, command group, command function and program length has been detected.

First Run

The date and time at which this command, with the same CICS APPLID, SYSID, transaction, program, offset, resource, remote SYSID, remote name, command group, command function and program length, was first detected. Times are shown in local time.

Last Run

The date and time at which this command, with the same CICS APPLID, SYSID, transaction, program, offset, resource, remote SYSID, remote name, command group, command function and program length, was first detected. Times are shown in local time.

Term Whether or not the transaction was associated with a terminal.

TCBmode

The CICS TCB mode at the time the command was executed.

DB2 report

The report lists all the DB2 commands that were collected by the Collector.

Figure 40 shows an example DB2 report.

```
2011/01/27 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320 - Page: 60
DB2 RESOURCES REPORT FOR APPLID: IYCYZC3C
```

TRAN	PROGRAM	OFFSET	COMMAND	TYPE	RESOURCE	DB2ID	PLAN			
			USAGE	FIRST RUN	LAST RUN	TERM	SECTION#	STATEMENT#	TCBMODE	
D8CS	DSN8CC0	000007E8	SELECT 25	2011-01-27 09.34.42	TABLE VCONA	2011-01-27 09.35.58	Y	1	DF2E 815	DSN8CC0 L8
	DSN8CC1	00002724	SELECT 25	2011-01-27 09.34.42	TABLE VCONA	2011-01-27 09.35.58	Y	4	DF2E 672	DSN8CC0 L8
		00002872	DELETE 4	2011-01-27 09.35.09	TABLE VCONA	2011-01-27 09.36.10	Y	5	DF2E 710	DSN8CC0 L8
		00002AEE	INSERT 4	2011-01-27 09.34.42	TABLE VCONA	2011-01-27 09.35.58	Y	6	DF2E 804	DSN8CC0 L8
		00002B98	UPDATE 17	2011-01-27 09.34.45	TABLE VCONA	2011-01-27 09.35.40	Y	7	DF2E 810	DSN8CC0 L8

Figure 40. Example DB2 report from the Dependency Reporter

The report fields contain the same information as the identically named fields on the CICS report, with the following additions and deletions:

DB2ID

The identifier (ID) of the DB2 subsystem.

PLAN The DB2 plan.

SECTION#

The section number in the source code of the CICS program at which the DB2 command is issued.

STATEMENT#

The statement number in the source code of the CICS program at which the DB2 command is issued.

TYPE The type of resource against which the command was issued.

Sysid Not reported on the DB2 report.

MQ report

The report lists all the MQ commands that were collected by the Collector.

Figure 41 shows an example MQ report.

```

2011/01/27 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320 - Page: 63
MQ RESOURCES REPORT FOR APPLID: IYCYZC3C
-----
Tran  Program  Offset  Command          Resource
-----
                Usage  First Run          Last Run          Term  TCBmode
-----
MAIL  CSQ4CVD1  00004C4E  OPEN QUEUE      CSQ4SAMP.MAILMGR.DJWHITE
                1  2011-01-27  09.24.16  2011-01-27  09.24.16  Y    QR
                00004CDA  CLOSE QUEUE     CSQ4SAMP.MAILMGR.DJWHITE
                1  2011-01-27  09.24.58  2011-01-27  09.24.58  Y    QR
                CSQ4CVD2  00004806  GET QUEUE       CSQ4SAMP.MAILMGR.DJWHITE
                24 2011-01-27  09.24.18  2011-01-27  09.24.18  Y    QR
                CSQ4CVD3  00003BA6  GET QUEUE       CSQ4SAMP.MAILMGR.DJWHITE
                2  2011-01-27  09.24.19  2011-01-27  09.24.22  Y    QR

```

Figure 41. Example MQ report from the Dependency Reporter

The report fields contain the same information as the identically named fields on the CICS report, with the following variations and deletions:

Resource

The resource against which the command was issued, causing a dependency between it and the named program. For the MQ report, this resource is the fully-qualified queue name.

Sysid Not reported on the MQ report.

IMS report

The report lists all the IMS commands that were collected by the Collector.

Figure 42 on page 133 shows an example IMS report.

Tran	Program	Offset	Command	Resource	Usage	First Run	Last Run	Term	TCBmode
DLE1	DLE10001	0000034A	EXEC DLI SCHEDULE	MDLPSBC	3,462	2011-01-20 13.49.57	2011-01-20 13.51.06	Y	QR
		0000044A	EXEC DLI INSERT	MDLHDAM	2,009	2011-01-20 13.49.57	2011-01-20 13.51.06	Y	QR
		0000054A	EXEC DLI INSERT	MDLHDAM	1,473	2011-01-20 13.49.57	2011-01-20 13.51.05	Y	QR
		0000064A	EXEC DLI GU	MDLHDAM	2,006	2011-01-20 13.49.57	2011-01-20 13.51.06	Y	QR
		0000074A	EXEC DLI REPLACE	MDLHDAM	1,999	2011-01-20 13.49.57	2011-01-20 13.51.05	Y	QR

Figure 42. Example IMS report from the Dependency Reporter

The report fields contain the same information as the identically named fields on the CICS report, with the following variations and deletions:

Sysid Not reported on the IMS report.

Running the Affinities Reporter

The CICS IA Affinities Reporter can be used to produce reports and definitions.

You can produce the following:

- A report of the affinities found in your CICS region. The commands reported are those listed in “Commands monitored for potential affinities” on page 192.
- Definitions of the basic affinity transaction groups that correspond to the report. The definitions of the basic affinity transaction groups are suitable for input to the Builder.

You can run the Affinities Reporter against either of the following:

- The Affinities database using the CIUAFFRD job.
- The VSAM affinity data files using the CIUAFFRP job. This option is useful if you do not have DB2.

The CIUAFFRD and CIUAFFRP jobs produce the same kinds of output. If the Affinities database and the VSAM files contain the same affinity data, the output from the jobs is identical.

For information about interpreting the report output by the Affinities Reporter, see “Running the affinity report” on page 138.

This rest of this section contains:

- “Requesting a report from the Affinities Reporter” on page 134
- “Output from the Affinities Reporter” on page 134
- “Running the affinity report” on page 138
- “Compressing affinity data” on page 140

Requesting a report from the Affinities Reporter

To run the Affinities Reporter against the Affinity database objects, edit and run the CIUAFFRD job. To run the Affinities Reporter against the VSAM affinity data files, edit and run the CIUAFFRP job. Before running either of the jobs, change the values of the parameters listed below, as appropriate.

The changes you need to make to the job stream are similar to those described in Updating the Affinity database objects and are listed in the header of the JCL file:

- The JOB accounting parameters.
- The PARM parameter of the EXEC statement

For example:

```
REPORT EXEC PGM=CIUAFFR,PARM='WORSEN'
```

. Worsen specifies that the Reporter is to worsen transaction affinity relations for affinities on which the Collector has not detected at least 10 occurrences.

- The STEPLIB DD statement; specify the name of the CICS IA load library where you have installed the Affinities Reporter program, CIUAFFR.
- The APPLID DD statement; specify the APPLID of the CICS TS region, for which you want to receive a report.
- The CMDGRPS DD statement, specify the affinity command types you want to see in the report. Only those affinity types listed on this DD statement are shown in the report. You can specify any of the following affinity types, with each type on a separate line, starting in column one:

CANCEL	DISCARD	GETMAIN	RESYNC
COLLECT	ENABLE	INQUIRE	RETRIEVE
CREATE	ENQ	LOAD	TS
CWA	EXTRACT	PERFORM	WAIT

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 affinity types selected.

- The TRANGRPS DD statement; specify the name of the sequential data set where the basic affinity-transaction-group definitions are to be sent.
- The SYSPRINT DD statement; specify the destination for the report.

Output from the Affinities Reporter

Each affinity type specified on the CMDGRPS DD statement, for example, CWA or TS is given its own section in the affinity report.

In each affinity-type section, there is an entry for each individual affinity of that type. And for each affinity, the Affinities Reporter creates a basic affinity-transaction-group definition that is suitable for input to the Builder.

Note:

1. The Affinities Reporter produces basic affinity-transaction-group definitions for *inter-transaction* affinities only.
2. Transactions not initiated from a terminal, not associated with a CBTS process or activity, and not associated with Link3270 bridge facilities, do not appear in a basic affinity transaction group. If none of the transactions in an inter-transaction affinity group are initiated from a terminal, are not associated with a CBTS process or activity, and are not

associated with Link3270 bridge facilities, a special reporting affinity relation of background is used; no basic affinity transaction group is created, ignore the affinity lifetime.

Affinity report

A sample report output by the Affinities Reporter.

The sample report below shows an example report for two affinities, a TS queue affinity and a CWA affinity. Only those affinity types were selected, as shown.

```

CICS INTERDEPENDENCY ANALYZER                                2007/09/24 Page 1
AFFINITY TYPE REPORTING OPTIONS                               Applid=CICSPDN1
Affinity Type      Reporting      Message
-----
1 Inter-Transaction Affinities 2
-----
CWA                Yes
CANCEL             No
ENQ                No
GETMAIN            No
LOAD               No
RETRIEVE           No
TS                 Yes
Transaction-System Affinities
-----
COLLECT            No
DISCARD            No
ENABLE             No
EXTRACT            No
INQUIRE           No
PERFORM            No
RESYNC             No
WAIT               No
CREATE             No
  
```

```

CICS INTERDEPENDENCY ANALYZER                                2007/09/24 Page 2 3
INTER-TRANSACTION AFFINITIES REPORT FOR ADDRESS CWA          Applid=CICSPDN1
Trangroup      : CW.00000001
Affinity       : GLOBAL
Lifetime       : SYSTEM
Tranid         Program      Offset      Usage      Command      Terminal    CBTS Task    Link3270
-----
AUXX           AUXXTST    000000CC    1          ADDRESS CWA   Yes         No          No
CWA1           AUCWA     FFFFFFFF    2          Yes          Yes         No          No
Total Transactions : 2
Total Programs    : 2
  
```

Figure 43. A sample report output by the Affinities Reporter

Notes for the sample report output by the Affinities Reporter, above:

1 Incorrect affinity types

This column lists any affinity types that were specified incorrectly on the CMDGRPS DD statement of the CIUAFFRD or CIUAFFRP job.

2 Affinity types reported

This column lists any affinity types that were selected for reporting; that is, those affinity types specified correctly on the CMDGRPS DD statement of the

CIUAFFRD or CIUAFFRP job. The affinity types are listed under their associated affinity category: inter-transaction or transaction-system.

3 Affinities reports

For each affinity transaction group, a report lists appropriate characteristics of the affinities, explained in the following notes.

Trangroup

The name of the affinity transaction group, assigned by the Affinities Reporter. This name is used only to cross-reference the group to the corresponding affinity-transaction-group definition in the data set specified on the TRANGRPS DD statement, for this run of the Affinities Reporter. The Trangroup value for an affinity transaction group might vary from one run to another of the Affinities Reporter.

Affinity

The affinity relation. If appropriate, this entry also indicates whether the relation was worsened from a less restrictive relation.

Lifetime

The affinity lifetime. If appropriate, this entry also indicates whether the lifetime was worsened from a less restrictive lifetime.

Queue (resource)

The resource causing the affinity. This entry might be the name of the resource, for example, Queue : LOCA1 (D7D6C3C1F140404040404040404040) as shown, or the address of the resource, depending on the type of affinity. An unprintable character appears as a period (.).

Recoverable

Whether or not the resource is recoverable. For TS queues, this entry also indicates whether the queue is in auxiliary or main temporary storage.

Terminal Id

The identifier of the terminal at which the transactions taking part in the affinity were initiated. This information is available only for TS queue affinity and is meaningful only if the affinity is LUNAME or worsened from LUNAME to GLOBAL. Therefore, the terminal identifier is included in the report only in these cases.

Tranid The identifier of each transaction taking part in the affinity. An affinity transaction group can contain only one transaction ID. An example of such a situation is when each part of a pseudoconversation accesses a TS queue and each part runs under the same transaction ID.

Program

The name of each program taking part in the affinity.

Offset The offset from the load point of the BALR instruction to the EXEC CICS command causing the affinity. The Affinities Reporter produces a negative offset (X'FFFFxxx') if it cannot determine an offset; that is, if the offset calculated is not within the program. A negative offset might indicate that the program, or perhaps language runtime code has passed control to another program by using a non-CICS mechanism; for example, a COBOL dynamic call.

This offset is not the same as the offset given by the Load Module Scanner, which is the offset of the command argument 0 declaration from the start of the load module.

If a negative offset (X'FFFFxxxx') is given, individual affinity commands cannot be directly located within a program. The program must be scanned for every instance of the affinity command, because there might be more than one.

Usage The number of times that this particular EXEC CICS command, with the transaction, program, and offset values reported, takes part in the affinity, up to a limit of 5000.

The usage count is an indication of the relative importance of the affinity. It is not a completely accurate usage count. For performance reasons, when the usage count is incremented by the Collector, the "save to file" flag is not necessarily set to indicate that the record needs to be saved to the data file. The save flag is set as follows:

0	<= usage count < 10,	save flag set every increment
10	<= usage count < 100,	save flag set every 10 increments
100	<= usage count < 5000,	save flag set every 100 increments
5000	<= usage count	neither increment nor save flag set

If the usage count is 1+, at least one example of the affinity was seen but the total number of occurrences of that affinity is unknown.

Command

The EXEC CICS command causing the affinity.

Terminal

Whether this particular EXEC CICS command, with the transaction, program, and offset values reported, was ever issued by a transaction initiated from a terminal; that is, started as a result of terminal input or for an EXEC CICS RETURN TRANSID command. ATI-started transactions are not included.

The word Mix in this column indicates that a particular EXEC CICS command was issued by a transaction initiated from a terminal and also issued by the transaction when it was initiated with no associated terminal.

CBTS Task

Whether this particular EXEC CICS command, with the transaction, program, and offset values reported, was ever issued by a BTS task.

Link3270

Whether this particular EXEC CICS command, with the transaction, program, and offset values reported, was ever issued by a transaction initiated by the Link3270 bridge mechanism.

Total Transactions

The total number of different transactions in the affinity transaction group.

Total Programs

The total number of different programs in the affinity transaction group.

Producing affinity-transaction-group definitions

The Affinities Reporter produces affinity-transaction-group definitions suitable for input to the Builder, but not to CICSplex SM.

Each definition consists of a unique transaction group name, a relation, a lifetime, and a set of transaction IDs (tranids).

Not every affinity in the report appears as an affinity transaction group. In particular, transaction-system affinities do not appear, because they are not of

interest to a dynamic routing program; nor, for the same reason, do transactions that were not initiated from a terminal, nor by BTS, nor by the Link3270 bridge mechanism.

Figure 44 shows a sample set of definitions to match the report in Figure 43 on page 135.

Note:

1. The transaction group name is not a valid CICSplex SM transaction group name, because the latter must be eight characters or less; it is used only as a cross-reference to the report.
2. MATCH or STATE attributes are not generated on CREATE TRANGRP commands, because those attributes are relevant only to the combined affinity transaction groups.
3. The HEADER statement is generated so that the Builder can detect a new data set in its input concatenation. It also gives the CICS APPLID and the date and time of the last Collector save. For more information see “HEADER statements” on page 162.

```
* HEADER APPLID(CICSPDN1)  SAVEDATE(2007/09/24)  SAVETIME(10:11:45);
*
* Generated by the CICS Interdependency Analyzer Reporter on 2007/09/24
* 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(AUX);
CREATE DTRINGRP TRANGRP(CW.00000001) TRANID(CWA1);
*
CREATE TRANGRP NAME(TS.00000001) AFFINITY(LUNAME ) AFFLIFE(PCONV )
      DESC(TS.LOCA1      D7D6C3C1F140404040404040404040);
CREATE DTRINGRP TRANGRP(TS.00000001) TRANID(AFTD);
CREATE DTRINGRP TRANGRP(TS.00000001) TRANID(AFTR);
CREATE DTRINGRP TRANGRP(TS.00000001) TRANID(AFTW);
*
CREATE TRANGRP NAME(TS.00000002) AFFINITY(LUNAME ) AFFLIFE(PCONV )
      DESC(TS.LOCA2      D7D6C3C1F240404040404040404040);
CREATE DTRINGRP TRANGRP(TS.00000002) TRANID(AFTD);
CREATE DTRINGRP TRANGRP(TS.00000002) TRANID(AFTR);
CREATE DTRINGRP TRANGRP(TS.00000002) TRANID(AFTW);
*
CREATE TRANGRP NAME(TS.00000003) AFFINITY(LINK3270) AFFLIFE(FACILITY )
      DESC(TS.TS_AFFINITY E3E26DC1C6C6C9D5C9E3E84040404040);
CREATE DTRINGRP TRANGRP(TS.00000003) TRANID(TSW1);
```

Figure 44. Sample basic affinity-transaction-group definitions

After these definitions have been created, you can edit them to add extra definitions for affinities that the Collector could not detect, or to modify definitions in the light of further knowledge about the affinity; for example, to correct a worsened lifetime. The report output from the Load Module Scanner might be particularly useful at this stage. See “Running the affinity report.”

Running the affinity report

Purpose of the report and assumptions to make.

The affinity report has two main purposes:

- To help you understand the affinities present in the CICS region concerned.

- To help you modify the affinity transaction-group-definitions before they are given to the Builder, if such modification is required.

You must be able to investigate whether any application changes could reduce the amount of affinity.

- Assume that the affinity information is complete.
- Assume that any worsening of affinity relation or affinity lifetime by the Collector does not create too pervasive an affinity, making dynamic routing less effective.

Understanding the affinities

The inter-transaction affinities listed in the report highlight those transactions that have affinities with other transactions.

Understanding the affinities present in the CICS region enables you to determine which of the them are most pervasive. If you decide that it is worth changing your application programs, it is generally more cost-effective to remove the most pervasive affinities, because those affinities most restrict dynamic routing. The most pervasive affinities are those with a relation of GLOBAL, or a lifetime of SYSTEM or PERMANENT, and are heavily used.

The transaction-system affinities listed in the report highlight those transactions that use system programming commands. It might not be appropriate to dynamically route such a transaction, because its action might be tied to a particular CICS region, as opposed to a particular set of other transactions.

The affinity report also lists affinities occurring between transactions that were not initiated from a terminal and are not BTS or Link3270 bridge transactions. These background transactions are known as “background relations”. This information is really for completeness, because such transactions cannot be dynamically routed.

To obtain complete information on affinities, use as many code paths as possible while running the Collector, because it can find an affinity only if the commands that cause it have been executed. However, the Load Module Scanner detects all instances of affinity commands in the corresponding load library. So a comparison of the Affinities Reporter and Load Module Scanner outputs is very useful when establishing the full picture.

Important note

Both the Affinities Reporter and the Load Module Scanner might identify commands that, on closer examination, do not cause real affinities. Relate the output from the Reporter and the Load Module Scanner to your knowledge of your applications, to distinguish between such commands and those causing real affinities that impact CICS dynamic routing.

Modifying affinity-transaction-group definitions

Modifications to consider before providing affinity-transaction-group definitions to the Builder.

- Remove any false affinities that might arise because the sharing of a resource is done on a read-only basis, making it 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 afterward. An alternative way to remove this false affinity is to prohibit detection of ADDRESS CWA by the Collector.

- Remove affinity relation worsening. An affinity that has a relation of LUNAME, BAPPL, LINK3270, or user ID might be worsened to GLOBAL because the Collector has not seen enough examples of the affinity to be convinced that it is related to a terminal, user ID, a BTS process or activity, or a Link3270 bridge. Change it to LUNAME, USERID, BAPPL, or LINK3270, and correct the lifetime, if you know that the affinity really is related to a terminal, user ID, a BTS process or activity, or a Link3270 bridge facility. You might want to prevent worsening by specifying WORSEN=NO.
- Remove affinity lifetime worsening; an LUNAME affinity with a lifetime of LOGON, or a USERID affinity with a lifetime of SIGNON, might be worsened to SYSTEM or PERMANENT because the Collector cannot always observe log offs or signoffs. Change this to LOGON or SIGNON if you know that to be the correct lifetime.
- You can change LUNAME affinity relation to USERID. An LUNAME affinity group might be both LUNAME and USERID, because all instances of all transactions in the group were initiated from the same terminal by the same USERID. This affinity group appears in the report as LUNAME, because LUNAME takes precedence. If you know that the affinity is primarily USERID related, change the affinity to USERID. This affinity might be indicated by other, similar, affinity groups appearing in the report with USERID.
- You can add WAIT affinities. The Affinities Reporter reports the use of WAIT EVENT, WAITCICS, and WAIT EXTERNAL commands as transaction-system affinities, because the Collector cannot detect the corresponding posting of the ECBs being waited on. Identify the posting transactions and create affinity transaction groups to describe the affinities. The output from the Load Module Scanner might be particularly useful here, because it finds programs that issue MVS POST commands.
- You can add other affinities. Load Module Scanner output or your knowledge of your applications might identify additional affinities. Create affinity transaction groups to describe them.
- You can add GETMAIN storage sharers. The Collector cannot detect transactions that share storage other than by EXEC CICS commands. Although it detects GETMAIN SHARED and FREEMAIN affinities, the address of the storage might have been passed to a third transaction. Add such transactions to the affinity transaction group.

Compressing affinity data

If your temporary storage queue names contain a unique counter or a terminal name (termid), a very large number of basic affinity transaction groups might be created for what might seem to be a small number of logical queues.

For example, consider the queues ABCD0001 through ABCD1000, with the name comprising a fixed part (ABCD) and a counter 0001 through 1000. They might result in 1000 basic affinity transaction groups, each with relation, LUNAME, lifetime PCONV, and transactions ABCD and ABCE. These groups form one logical queue, ABCD*, which causes an affinity that might be described by one affinity transaction group. However, the result is 1000 basic affinity transaction groups.

The affinity data might be more readable if compressed to its logical form. Use the Builder to do this, because it combines all affinity transaction groups that contain the same transaction ID. The Builder output for the previous example would be one affinity transaction group with relation LUNAME, lifetime PCONV, and transactions ABCD and ABCE.

Chapter 9. Running the Program Threadsafe report

This section describes the CICS IA program dynamic analysis threadsafe report. The report consists of batch job CIUJTSQ2, which produces reports displaying the threadsafe status of each command in a program.

The threadsafe status for a command can be as follows:

Threadsafe

An EXEC CICS command that does not cause a TCB swap.

Non-Threadsafe

An EXEC CICS command that can cause a TCB swap.

Indeterminate Threadsafe

An EXEC CICS command where it cannot be determined if the call causes a TCB swap.

Dynamic call

A call to another module a execution time. The call was not initiated using an EXEC CICS command.

Threadsafe Inhibitor call

An EXEC CICS command that can cause an unsafe affinity between transactions. You need to investigate the call to determine if it inhibits the program from being threadsafe. These commands are: ADDRESS CWA, LOAD HOLD, GETMAIN SHARED, and EXTRACT EXIT.

DB2 calls

The calls to the CICS DB2 interface are threadsafe.

IMS calls

The calls to the CICS IMS interface are non-threadsaf.

MQ calls

The calls to the CICS MQ interface are threadsafe only in CICS TS V3.2.

To request a dynamic analysis threadsafe report, edit and run the CIUJTSQ2 job.

Analyzing the program dynamic analysis threadsafe report

A sample of part of the report create by the program dynamic analysis threadsafe report program.

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

Report options:
 PROGRAMNAME=** REGIONNAME=** CICSLEVEL= REPORT=DETAIL LINESPERPAGE=60

Definitions of Terms:

- 'Threadsafe' calls are EXEC CALLS commands that do not cause a TCB swap.
- 'Non-Threadsafe' calls are EXEC CALLS commands that cause a TCB swap.
- 'Indeterminate Threadsafe' calls are EXEC CALLS commands where it cannot be determined if the call causes a TCB swap.
- 'Dynamic calls' are calls to modules at execution time. Programs that are called dynamically take on the same environment as the calling program.
- 'Threadsafe Inhibitor calls' are EXEC CICS commands that need to be investigated further because they may prevent you from defining your program as threadsafe. These commands are: ADDRESS CWA, EXTRACT EXIT, GETMAIN SHARED, and LOAD.

Figure 45. Example program dynamic analysis threadsafe report, header page

APPLID	Program	Linkedit Date	Execution Key	Concurrency	APIST	Storage Protect	CICS Rel	LIB Dataset Name				
	CMD Type	Function	Type	Resource	Offset	Program Length	Use Count	Threadsafe				
IYDZZ322	DCR78001	-----	USER	QUASIRENT	CICSAPI	ACTIVE	0650	CICSIAD.IA2.TEST.LOADLIB		1		
	CICS CREATE			IPCONN	IPCONN01		602	1810	1	N		
	CICS CREATE			LIBRARY	TESTLIBR		7DE	1810	1	N		
	CICS DISCARD			IPCONN	IPCONN01		764	1810	1	N		
	CICS DISCARD			LIBRARY	TESTLIBR		904	1810	1	N	2	
	CICS INQUIRE			IPCONN	IPCONN01		72A	1810	1	Y		
	CICS INQUIRE			LIBRARY	TESTLIBR		8CA	1810	1	Y		
	CICS SET			IPCONN	IPCONN01		6E8	1810	1	Y		
	CICS SET			LIBRARY	TESTLIBR		834	1810	1	Y		
Total CICS calls:		8	Threadsafe:	4	Non-Threadsafe:	4	Indeterminate	Threadsafe:	0	3		
			DB2 calls:	0	MQ calls:	0	IMS calls:		0			
			Dynamic Calls:	0	Threadsafe Inhibitor calls:	0						

Figure 46. Example programs dynamic analysis threadsafe report, main body

1 For the detail and summary report, the programs requested by the report options PROGRAMNAME and REGIONNAME are listed. These program entries contain the following information:

APPLID

The application ID for the CICS region from which the Collector captured the program information. This field matches the report criteria specified by the REGIONNAME report option.

Program

The name of the program for which the information is reported.

Linkedit Date

The linkedit date of the program.

Execution Key

The storage key of the program. Values are CICS, USER, and NOTAPPLIC.

Concurrency

Indicates the concurrency attribute of the installed program definition. Values are: QUASIRENT and THREADSAFE.

APIST

Indicates the API attribute of the installed program definition. Values are: CICSAPI and OPENAPI.

Storage Protect

Indicates if storage protection was active for the program. Values are: ACTIVE or INACTIVE.

CICS Rel

The CICS release number for the region in which the program is running.

LIB Dataset Name

The 44-character name of the data set from which the program was loaded into the CICS region.

2 For the detail report, all of the commands that were collected by the Collector for each program are listed. These command entries contain the following information:

CMD Type

The type of command invoked by the program. Values are: CICS, DB2, IMS, and MQ.

Function

The command function, as specified in the CIU_CICS_DATA table.

Type The resource type such as TS, or Program, as specified in the CIU_CICS_DATA table.

Resource

The name of the resource that the command was acting upon, as specified in the CIU_CICS_DATA table.

Offset The offset of the command from the start of the program module.

Program Length

The length of the program module. Used to help determine the program version.

Use Count

The number of times that the command was run.

Threadsafe

The threadsafe status of the command. Values are:

Y The command is threadsafe.

N The command is not threadsafe.

I The threadsafe status of the command is indeterminate. More investigation is needed to determine if the command is threadsafe.

3 A summary of the types of commands issued by the program are listed after each program entry.

CICS calls

The number of **EXEC CICS** commands invoked by the program.

Threadsafe

The number of **EXEC CICS** commands invoked by the program that are threadsafe.

Non-Threadsafe

The number of **EXEC CICS** commands invoked by the program that are not threadsafe.

Indeterminate Threadsafe

The number of **EXEC CICS** commands invoked by the program that cannot be determined to be threadsafe or not.

DB2 calls

The number of DB2 commands invoked by the program.

MQ calls

The number of MQ commands invoked by the program.

IMS calls

The number of IMS commands invoked by the program.

Dynamic calls

The number of calls made to other modules by the program.

Threadsafesafe Inhibitor calls

The number of **EXEC CICS** commands invoked by the program for which you need to investigate the command further to determine if it prevents the program from being threadsafe.

Chapter 10. Running the Load Module Scanner

The Load Module Scanner scans load modules for instances of program commands that could cause resource dependencies or transaction affinities.

This section describes how to run the CICS IA Load Module Scanner. The Load Module Scanner works by scanning the load modules for patterns of bits that might be commands.

The Load Module Scanner detects the use of:

- The dependency-related commands listed in “Dependency-related commands” on page 6
- The affinity-related EXEC CICS API and SPI commands 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 use the Load Module Scanner to obtain any of the following:

- A summary printed report listing the total number of modules scanned, the total that contain possible dependency-causing or affinity-causing commands, and similar conditions by running the CIUJCLLS job.
- A summary printed report as above, with a separate list of modules that contain possible dependency-causing or affinity-causing commands, for input to a further job to produce a detailed report by running the CIUJCLLS job.
- A summary printed report as above, with or without a separate module list, with updates to the CIU_SCAN_SUMMARY DB2 table by running the CIUJCLTS job. Your own programs can process the scan results by querying the CIU_SCAN_SUMMARY table.
- A detailed printed report listing each possible dependency-causing or affinity-causing command in the scanned modules, with further information about the command by running the CIUJCLLD job.
- A detailed printed report as above, with updates to the CIU_SCAN_DETAIL DB2 table by running the CIUJCLTD job. Your own programs can process the scan results by querying the CIU_SCAN_DETAIL table.

You are recommended to use the Load Module Scanner by:

1. Create a summary report and module list to identify modules that contain possible dependency-causing or affinity-causing commands. See “Creating a summary report.”
2. Produce detailed reports to review modules that the summary report has identified. See “Creating a detailed report” on page 148.

Creating a summary report

You can request a summary report from the Load Module Scanner by editing and running the CIUJCLLS job.

This job can also produce a separate list of modules that contain possible dependency-causing or affinity-causing commands, for input to the CIUJCLLD or CIUJCLTD job for more detailed reporting.

Before running the CIUJCLLS job, change the following as appropriate:

- The JOB accounting parameters
- The PARM keyword of the EXEC statement:

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

where:

gjSUMMARY

Specifies that a summary scan and report is required for the entire library, except for CICS modules, CICS IA modules, CICS tables, and those modules that cannot be loaded, because of an error.

DETAILMODS

Specifies that the names of those modules containing at least one possible dependency-causing or affinity-causing command are to be written to the sequential file defined by the INTMOD DD statement. This file might be used to restrict a subsequent detailed report, by specifying it on the DETAIL DD statement of a detailed report run of the Load Module Scanner.

- The STEPLIB DD statement; specify the name of the CICS IA load library in which you have installed the Load Module Scanner program, CIULMS. The default is hlq.SCIULOAD, where hlq is the data set qualifier assigned during installation.
- The INPUT DD statement; specify the name of the load library to be scanned.
- The SYSPRINT DD statement; specify the destination for the summary report.
- The INTMOD DD statement; specify 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 running the Load Module Scanner to produce a detailed report.
- The CIUPRINT DD statement; specify the destination of error messages (SYSOUT=*).

You do not need the DETAIL DD statement, dummy, for a summary run.

Each summary report contains:

- A separate line giving the following information about each module in the library:
 - Name
 - Size
 - Language (if determined)
 - Language version: Language Environment (LE) or non-LE
 - Number of possible affinity-causing commands
 - Number of possible dependency-causing commands
 - Number of possible MVS POST commands

Note: If a load module is created from several source languages, only one language is indicated.

- The total count of:
 - Modules in the library
 - Modules scanned
 - CICS modules and tables (not scanned)
 - Modules in error (not scanned)
 - Modules that contain MVS POST commands
 - Modules that contain commands that might cause dependencies

- Modules that contain commands that might cause affinities
- Assembler modules
- C modules
- COBOL modules
- PL/I modules

Figure 47 is an example of a summary report produced by the Load Module Scanner.

CICS INTERDEPENDENCY ANALYZER Version 2.2.0 09/19/07 Page 1
 LOAD MODULE SCANNER - SUMMARY LISTING OF CICSTLS.CICS1104.LOADLIB

Module Name	Module Length	Module Language	Language Version	Possible statements Affinities	Dependencies	MVS POSTs	Comment
DFHPLTPI		CICS TABLE					
DFHPLTSD		CICS TABLE					
EMSTESTA	00001FA8	COBOL II	LE	14	14	0	
EMSTESTB	000014A8	COBOL II	LE	0	0	0	
EMSTESTC	00000A48	COBOL II	Non LE	3	3	0	

CICS INTERDEPENDENCY ANALYZER Version 2.2.0 09/19/07 Page 2
 LOAD MODULE SCANNER - SUMMARY LISTING OF CICSTLS.CICS1104.LOADLIB

LOAD LIBRARY STATISTICS

```

=====
Total modules in library           = 5
Total modules scanned              = 3
Total CICS modules/tables (not scanned) = 2
Total modules in error (not scanned) = 0
Total modules containing possible MVS POSTs = 0
Total modules containing possible Dependency commands = 2
Total modules containing possible Affinity commands = 2
Total ASSEMBLER modules           = 0
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 = 17
Total number of possible Affinity commands = 17
  
```

Figure 47. Example of a summary report produced by the Load Module Scanner

Creating a summary report with DB2 output

You can request a summary report, with DB2 output, by editing and running the CIUJCLTS job.

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.

Like CIUJCLLS, the CIUJCLTS job can also produce a separate list of modules that contain possible dependency-causing or affinity-causing commands, for input to the CIUJCLLD or CIUJCLTD job for more detailed reporting.

Before running the CIUJCLTS job, change the following:

- The JOB accounting parameters

- The PARM keyword of the EXEC statement:

PARM= ' SUMMARY [,DETAILMODS] [, TABLE] '

where:

SUMMARY

Specifies that a summary scan, and report, is required for the entire library, except for CICS modules, CICS IA modules, CICS tables, and those modules that cannot be loaded, because of an error.

DETAILMODS

Specifies that the names of those modules containing at least one possible dependency-causing or affinity-causing command are to be written to the sequential file defined by the INTMOD DD statement. This file might be used to restrict a subsequent detailed report, by specifying it on the DETAIL DD statement of a detailed report run of the Load Module Scanner.

TABLE

Specifies that the results of the summary scan are to be written to the DB2 table CIU_SCAN_SUMMARY.

- The SYS keyword of the PARM keyword of the EXEC statement; specify the name of the DB2 subsystem.
- The STEPLIB DD statement; specify the name of the CICS IA load library in which you have installed the Load Module Scanner program, CIULMS. The default is hlq.SCIULOAD, where hlq is the 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 data set qualifier assigned to the DB2 subsystem during installation.
- The INPUT DD statement; specify the name of the load library to be scanned.
- The SYSPRINT DD statement; specify the destination for the summary report.
- The SYSPROC DD statement; specify the name of the CICS IA CLIST library. The default is hlq.SCIUCLIS, where hlq is the data set qualifier assigned during installation.
- The INTMOD DD statement; specify 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 running the Load Module Scanner to produce a detailed report.

You do not need the DETAIL DD statement, dummy, for a summary run.

Creating a detailed report

You can request a detailed report from the Load Module Scanner by editing and running the job CIUJCLLD.

Change the following statements as appropriate:

- The JOB accounting parameters
- The PARM keyword of the EXEC statement

PARM= ' DETAIL [,ALL] '

where:

DETAIL

Specifies that a detailed scan and report is 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 and affinity-causing commands.

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

- The STEPLIB DD statement, specify the name of the CICS IA load library in which you have installed the Load Module Scanner program, CIULMS. The default is hlq.SCIULOAD.
- The INPUT DD statement, specify the name of the load library to be scanned.
- The SYSPRINT DD statement, specify the destination for the detailed report.
- The DETAIL DD statement, specify the name of the data set containing the list of modules to be scanned. This list might be created initially as the output from a summary run of the Load Module Scanner. If you specify ALL on the PARM statement, change the DETAIL DD statement to specify //DETAIL DD DUMMY.
- CIUPRINT DD statement; specify the destination of error messages (SYSOUT=*).

You do not need the INTMOD DD, dummy, statement for a detailed run.

Contents of a detailed report

An example of a detailed report produced by the Load Module Scanner.

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

- A header line giving the name, size, and entry point of the module.
- A line for each possible dependency-causing, affinity-causing, and MVS POST command found, giving:
 - The offset of the command argument zero declaration from the start of the load module. This offset is not the same as the offset given by the Reporter; the offset given by the Reporter is for the command itself.
 - The contents of the command argument zero declaration, in hexadecimal.
 - The EDF DEBUG line number, if present. The line number 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, the absence of a DEBUG line number might indicate that what was found was not an argument zero.
 - What the command appears to be, for example, WRITEQ TS.
 - Whether the command appears to be a dependency-causing command.
 - The type of affinity, if any, that might be caused by this command.
- A summary report of the modules, giving:
 - The total possible affinity-causing commands
 - The total possible dependency-causing commands
 - The total possible MVS POST commands
- Library totals, as for the summary report, but only for those modules selected for the detailed run.

An example of a detailed report produced by the Load Module Scanner is below:

Module Name - DB010001 / Load Module Length - 00002E00	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
000002B3 0802E0000700004000	00771 WRITEQ TD Yes	
000002D5 0E0280002700000100	00668 LINK PROGRAM Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	2	
Total possible MVS POSTs =	0	

Module Name - DB900001 / Load Module Length - 00002130	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
000001EF 1804F100070000000015E204000020	00561 SEND MAP Yes	
00000206 1804F100070000000015E204000020	00451 SEND MAP Yes	
0000021D 1802D0000700000000050900000020	00346 RECEIVE MAP Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	3	
Total possible MVS POSTs =	0	

Module Name - DB910001 / Load Module Length - 00003250	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
00000223 1804F100070000000015E204000020	01140 SEND MAP Yes	
0000023A 1804F100070000000015E204000020	01015 SEND MAP Yes	
00000251 1802D0000700000000050900000020	00729 RECEIVE MAP Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	3	
Total possible MVS POSTs =	0	

Module Name - DB920001 / Load Module Length - 00003588	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
0000025A 1804F100070000000015E204000020	01236 SEND MAP Yes	
00000271 1804F100070000000015E204000020	01100 SEND MAP Yes	
00000288 1802D0000700000000050900000020	00774 RECEIVE MAP Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	3	
Total possible MVS POSTs =	0	

Module Name - DB930001 / Load Module Length - 00001FB0	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
000001E5 1804F100070000000015E204000020	00503 SEND MAP Yes	
000001FC 1804F100070000000015E204000020	00390 SEND MAP Yes	
00000213 1802D0000700000000050900000020	00312 RECEIVE MAP Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	3	
Total possible MVS POSTs =	0	

Module Name - DB940001 / Load Module Length - 00002648	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	
0000020F 1804F100070000000015E204000020	00759 SEND MAP Yes	
00000226 1804F100070000000015E204000020	00644 SEND MAP Yes	
0000023D 1802D0000700000000050900000020	00505 RECEIVE MAP Yes	
Total possible Affinity commands =	0	
Total possible Dependency commands =	3	
Total possible MVS POSTs =	0	

Module Name - DB950001 / Load Module Length - 00003EE8	Module Entry Point - 00000020	Affinity
Offset Storage Content (HEX)	EDF DEBUG Possible Command Depcy	

```

-----
00000255 1804F100070000000015E204000020          02042 SEND MAP          Yes
0000026C 1804F100070000000015E204000020          01919 SEND MAP          Yes
00000283 1802D0000700000000050900000020          01573 RECEIVE MAP       Yes
Total possible Affinity commands =          0
Total possible Dependency commands =          3
Total possible MVS POSTs =          0

```

```

Module Name - DB960001 / Load Module Length - 00003EC0 / Module Entry Point - 00000020
Offset      Storage Content (HEX)                EDF DEBUG Possible Command  Depcy  Affinity
-----
00000255 1804F100070000000015E204000020          02028 SEND MAP          Yes
0000026C 1804F100070000000015E204000020          01905 SEND MAP          Yes
00000283 1802D0000700000000050900000020          01560 RECEIVE MAP       Yes
Total possible Affinity commands =          0
Total possible Dependency commands =          3
Total possible MVS POSTs =          0

```

I CICS INTERDEPENDENCY ANALYZER Version 3.2.0 09/01/11 Page 3
LOAD MODULE SCANNER - DETAILED LISTING OF CICSTLS.STRESS.LOADLIB

```

Module Name - RDORCTTS / Load Module Length - 00002868 / Module Entry Point - 00000028
Offset      Storage Content (HEX)                EDF DEBUG Possible Command  Depcy  Affinity
-----
0000028A 0A02E8000700004180          00045 WRITEQ TS          Yes    Trans
000002AC 0802E0000700004000          00040 WRITEQ TD          Yes
000002BD 0802E0000700004000          00040 WRITEQ TD          Yes
000002CE 0802E0000700004000          00039 WRITEQ TD          Yes
000002DF 0802E0000700004000          00039 WRITEQ TD          Yes
000002F0 0802E0000700004000          00037 WRITEQ TD          Yes
00000301 0A0680000700002100          00029 DELETEQ TS         Yes    Trans
00000323 0E0280002700000100          00024 LINK PROGRA       Yes
Total possible Affinity commands =          2
Total possible Dependency commands =          8
Total possible MVS POSTs =          0

```

I CICS INTERDEPENDENCY ANALYZER Version 3.2.0 09/01/11 Page 4
LOAD MODULE SCANNER - DETAILED LISTING OF CICSTLS.STRESS.LOADLIB

LOAD LIBRARY STATISTICS

```

=====
Total modules in DETAIL file =          9
Total modules scanned =          9
Total CICS modules/tables (not scanned) =          0
Total modules in error (not scanned) =          0
Total modules containing possible MVS POSTs =          0
Total modules containing possible Dependency commands =          9
Total modules containing possible Affinity commands =          1
Total ASSEMBLER modules =          0
Total C/370 modules =          0
Total COBOL modules =          0
Total COBOL II modules =          9
Total PL/I modules =          0
Total number of possible Dependency commands =          26
Total number of possible Affinity commands =          2

```

Creating a detailed report with DB2 output

You can request a detailed report, with DB2 output, by editing and running the CIUJCLTD job.

The CIUJCLTD job updates the CIU_SCAN_DETAIL DB2 table with the results of the scan. Your own programs can then process the scan results by querying the CIU_SCAN_DETAIL table.

Before running the CIUJCLTD job, change the following:

- The JOB accounting parameters
- The PARM keyword of the EXEC statement

PARM= 'DETAIL[,ALL][,TABLE]'

where:

DETAIL

Specifies that a detailed scan and report is 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 and affinity-causing commands.

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

TABLE

Specifies that the results of the summary scan are to be written to the DB2 table CIU_SCAN_SUMMARY.

- The SYS keyword of the PARM keyword of the EXEC statement, specify the name of the DB2 subsystem.
- The STEPLIB DD statement, specify the name of the CICS IA load library in which you have 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, where db2hlq is the data set qualifier assigned to the DB2 subsystem during installation.

- The INPUT DD statement, specify the name of the load library to be scanned.
- The SYSPRINT DD statement, specify the destination for the detailed report.
- The SYSPROC DD statement, specify the name of the CICS IA CLIST library. The default is hlq.SCIUCLIS, where hlq is the data set qualifier assigned during installation.
- The DETAIL DD statement, specify the name of the data set containing the list of modules to be scanned. This list might be created initially as the output from a summary run of the Load Module Scanner. If you specify ALL on the PARM statement, change the DETAIL DD statement to specify //DETAIL DD DUMMY.

You do not require the INTMOD DD, dummy, statement for a detailed run.

Chapter 11. Running the CSECT Scanner

This section describes how to run the CICS IA CSECT Scanner. The CSECT Scanner scans load modules for information that can be used to identify the version of each CSECT.

The output is stored in DB2 tables and can be used, in conjunction with the DB2 dependency tables, to identify different versions of programs.

For each load module the CSECT Scanner records:

- Load module length
- Entry point offset
- AMODE
- RMODE
- Linkage editor or binder identifier and version
- Linkage edit or bind timestamp

For each CSECT the CSECT Scanner records:

- Translator (compiler) identifier and version
- Translation (compile) date
- User data specified on the Linkage Editor IDENTIFY control statement
- HMASPZAP data, specified during HMASPZAP processing

The Scanner generates a printed report. Optionally, the load module information is added to the CIU_PROGRAM_INFO DB2 table and the CSECT information is added to the CIU_CSECT_INFO table.

To help you identify translators and linkage editors, a third DB2 table, CIU_TRANSLATORS, is preloaded with information about the IBM assemblers, compilers, linkage editors, and binders that are likely to be found. You can use this table to convert translator identifiers into more easily understood descriptions.

The key items of information in the DB2 dependency tables that can be used to identify load module versions are program name and program length. This information is not ideal because the program length is not always unique for each version of a program; however, it is all that is available to the Collector at runtime. Also, the length of programs placed in a Partitioned Data Set Extended (PDSE) can be rounded up to multiples of 4 KB, to prevent this rounding specify the FETCHOPT(PACK,PRIME) option when link-editing the programs.

Run the CSECT Scanner whenever load libraries that contain programs for which interdependency data is being collected are changed. This run adds information about any new or changed programs and CSECTS to the DB2 tables. The structure of the tables produced by the CSECT Scanner is shown in “The structure of the CSECT Scanner objects” on page 219.

You can use the CSECT Scanner database objects with the Dependency database objects to compare the dependencies of different versions of a program. You can also use the two database objects to identify dependency data that applies only to an old version of a program, so that the data can be deleted.

The CIUJCLCS job

Run the CSECT Scanner by editing and running the CIUJCLCS job.

This job produces a printed report and updates the CIU_PROGRAM_INFO and CIU_CSECT_INFO DB2 tables.

Before running the CIUJCLCS job, change the following 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 PARM keyword of the EXEC statement; specify the name of the DB2 subsystem.
- The STEPLIB DD statement; specify 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; specify the name of the load library to be scanned.
- The SYSPRINT DD statement; specify the destination for the printed report.
- The SYSPROC DD statement; Specify the name of the CICS IA CLIST library. The default is hlq.SCIUCLIS, where hlq is the high-level data set qualifier assigned during installation.

Contents of the printed report

The printed report has two line formats, one for load module information and one for CSECT information.

The load module information consists of:

- Program (load module) name.
- Program length in hexadecimal.
- Entry point offset in hexadecimal.
- If the program name is an alias, the name of the program of which it is an alias. CSECT information is not listed for aliases.
- Binder or linkage editor identifier.
- Binder or linkage editor version.
- Bind or link-edit timestamp.
- Addressing mode (AMODE).
- Residency mode (RMODE).

The CSECT information consists of:

- CSECT name.
- First translation date.
- First translator identifier.
- First translator version.
- Second translation date.
- Second translator identifier.
- Second translator version.

- User data date.
- User data.
- HMASPZAP date.
- HMASPZAP data.

Below is an example of a report produced by the CSECT Scanner:

CICS INTERDEPENDENCY ANALYZER Version 3.2.0 10/01/11 Page 1
 CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE		
CSECT	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData	ZAPdate	ZAPdata
DB010001	00002E00	00000020		5695DF108	02.10	2004013144847	31	ANY		
DFHECI	2003318	569623400	01.04							
DB010001	2004013	5648A2500	21.00							
DSNCLI	2003318	569623400	01.04							
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198		
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04				
DSNHADD2	2002037	569623400	01.02				2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02							
DSNHMVHW	1998191	569623400	01.02							
CEEBETBL	2001115	569623400	01.04							
CEESTART	2001115	569623400	01.04							
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061		
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEEBPIRA	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEECPYRT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEEBPUBT	2001115	569623400	01.04							
CEEBTRM	2001115	569623400	01.04							
CEEBLLST	2001115	569623400	01.04							
CEEBINT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
DB900001	00002130	00000020		5695DF108	02.10	2004013145143	31	ANY		
DFHECI	2003318	569623400	01.04							
DB900001	2004013	5648A2500	21.00							
DSNCLI	2003318	569623400	01.04							
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198		
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04				
DSNHADD2	2002037	569623400	01.02				2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02							
DSNHMVHW	1998191	569623400	01.02							
CEEBETBL	2001115	569623400	01.04							
CEESTART	2001115	569623400	01.04							
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061		
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEEBPIRA	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEECPYRT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEEBPUBT	2001115	569623400	01.04							
CEEBTRM	2001115	569623400	01.04							
CEEBLLST	2001115	569623400	01.04							
CEEBINT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
DB910001	00003250	00000020		5695DF108	02.10	2004013145224	31	ANY		
DFHECI	2003318	569623400	01.04							
DB910001	2004013	5648A2500	21.00							
DSNCLI	2003318	569623400	01.04							
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198		
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04				
DSNHADD2	2002037	569623400	01.02				2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02							
DSNHMVHW	1998191	569623400	01.02							
CEEBETBL	2001115	569623400	01.04							
CEESTART	2001115	569623400	01.04							
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061		
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01				

CICS INTERDEPENDENCY ANALYZER Version 3.2.0 10/01/11 Page 2
 CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE		
CSECT	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData	ZAPdate	ZAPdata
CEEBPIRA	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEECPYRT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
CEEBPUBT	2001115	569623400	01.04							
CEEBTRM	2001115	569623400	01.04							
CEEBLLST	2001115	569623400	01.04							
CEEBINT	2001115	569623400	01.04	2001115	PL/X-390	02.01				
DB920001	00003588	00000020		5695DF108	02.10	2004013145237	31	ANY		
DFHECI	2003318	569623400	01.04							
DB920001	2004013	5648A2500	21.00							
DSNCLI	2003318	569623400	01.04							
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198		
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04				
DSNHADD2	2002037	569623400	01.02				2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02							

```

DSNHMVHW 1998191 569623400 01.02
CEEBETBL 2001115 569623400 01.04
CEESTART 2001115 569623400 01.04
IGZCBSO 2001115 569623400 01.04 2001115 PL/X-370 01.04 2001116 RS111151061
CEEARLU 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEEBPIRA 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEECPYRT 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEEBPUBT 2001115 569623400 01.04
CEEBTRM 2001115 569623400 01.04
CEEBLLST 2001115 569623400 01.04
CEEBINT 2001115 569623400 01.04 2001115 PL/X-390 02.01
DB930001 00001FB0 00000020 5695DF108 02.10 2004013145257 31 ANY
DFHECI 2003318 569623400 01.04
DB930001 2004013 5648A2500 21.00
DSNCLI 2003318 569623400 01.04
CEESG005 2001115 569623400 01.04 2001115 PL/X-370 01.04 2001116 RS111151198
DSNAA 1998295 569623400 01.02 1998295 PL/X-370 01.04
DSNHADD2 2002037 569623400 01.02 2002064 UQ62510
DSNHADDR 1998191 569623400 01.02
DSNHMVHW 1998191 569623400 01.02
CEEBETBL 2001115 569623400 01.04

CEESTART 2001115 569623400 01.04
IGZCBSO 2001115 569623400 01.04 2001115 PL/X-370 01.04 2001116 RS111151061
CEEARLU 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEEBPIRA 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEECPYRT 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEEBPUBT 2001115 569623400 01.04
CEEBTRM 2001115 569623400 01.04
CEEBLLST 2001115 569623400 01.04
CEEBINT 2001115 569623400 01.04 2001115 PL/X-390 02.01
DB940001 00002648 00000020 5695DF108 02.10 2004013145311 31 ANY
DFHECI 2003318 569623400 01.04
DB940001 2004013 5648A2500 21.00
DSNCLI 2003318 569623400 01.04
CEESG005 2001115 569623400 01.04 2001115 PL/X-370 01.04 2001116 RS111151198
DSNAA 1998295 569623400 01.02 1998295 PL/X-370 01.04
DSNHADD2 2002037 569623400 01.02 2002064 UQ62510

```

CICS INTERDEPENDENCY ANALYZER Version 3.2.0
CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

10/01/11 Page 3

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE	
CSECT	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData	ZAPdate ZAPdata
DSNHADDR	1998191	569623400	01.02						
DSNHMVHW	1998191	569623400	01.02						
CEEBETBL	2001115	569623400	01.04						
CEESTART	2001115	569623400	01.04						
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061	
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEEBPIRA	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEECPYRT	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEEBPUBT	2001115	569623400	01.04						
CEEBTRM	2001115	569623400	01.04						
CEEBLLST	2001115	569623400	01.04						
CEEBINT	2001115	569623400	01.04	2001115	PL/X-390	02.01			
DB950001	00003EE8	00000020		5695DF108	02.10	2004013145334	31	ANY	
DFHECI	2003318	569623400	01.04						
DB950001	2004013	5648A2500	21.00						
DSNCLI	2003318	569623400	01.04						
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198	
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04			
DSNHADD2	2002037	569623400	01.02			2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02						
DSNHMVHW	1998191	569623400	01.02						
CEEBETBL	2001115	569623400	01.04						
CEESTART	2001115	569623400	01.04						
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061	
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEEBPIRA	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEECPYRT	2001115	569623400	01.04	2001115	PL/X-390	02.01			
CEEBPUBT	2001115	569623400	01.04						
CEEBTRM	2001115	569623400	01.04						
CEEBLLST	2001115	569623400	01.04						
CEEBINT	2001115	569623400	01.04	2001115	PL/X-390	02.01			
DB960001	00003EC0	00000020		5695DF108	02.10	2004013145348	31	ANY	
DFHECI	2003318	569623400	01.04						
DB960001	2004013	5648A2500	21.00						
DSNCLI	2003318	569623400	01.04						
CEESG005	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151198	
DSNAA	1998295	569623400	01.02	1998295	PL/X-370	01.04			
DSNHADD2	2002037	569623400	01.02			2002064	UQ62510		
DSNHADDR	1998191	569623400	01.02						
DSNHMVHW	1998191	569623400	01.02						
CEEBETBL	2001115	569623400	01.04						
CEESTART	2001115	569623400	01.04						
IGZCBSO	2001115	569623400	01.04	2001115	PL/X-370	01.04	2001116	RS111151061	
CEEARLU	2001115	569623400	01.04	2001115	PL/X-390	02.01			

```

CEEBPIRA 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEECPYRT 2001115 569623400 01.04 2001115 PL/X-390 02.01
CEEBPUBT 2001115 569623400 01.04
CEEBTRM 2001115 569623400 01.04
CEEBLLST 2001115 569623400 01.04
CEEBINT 2001115 569623400 01.04 2001115 PL/X-390 02.01
RDORCTTS 00002868 00000028 5695DF108 02.10 2004013160928 31 ANY

```

CICS INTERDEPENDENCY ANALYZER Version 3.2.0

10/01/11 Page 4

CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE	ZAPdate	ZAPdata
CSECT	T1date	T1name	T1ver T2date	T2name	T2ver	UsrDate	UserData			
DFHELII	2003318	569623400	01.04							
RFWDB201	2004013	5648A2500	22.01							
DSNCLI	2003318	569623400	01.04							
CEESG005	2001115	569623400	01.04	2001115 PL/X-370	01.04	2001116	RSI111151198			
CEEBETBL	2001115	569623400	01.04							
CEESTART	2001115	569623400	01.04							
IGZCBSO	2001115	569623400	01.04	2001115 PL/X-370	01.04	2001116	RSI111151061			
CEEARLU	2001115	569623400	01.04	2001115 PL/X-390	02.01					
CEEBPIRA	2001115	569623400	01.04	2001115 PL/X-390	02.01					
CEECPYRT	2001115	569623400	01.04	2001115 PL/X-390	02.01					
CEEBPUBT	2001115	569623400	01.04							
CEEBTRM	2001115	569623400	01.04							
CEEBLLST	2001115	569623400	01.04							
CEEBINT	2001115	569623400	01.04	2001115 PL/X-390	02.01					

Chapter 12. Running the Builder

The CICS IA Builder runs as a batch job to build affinity-transaction-group definitions suitable for input to the CICS system management product, CICSplex SM.

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

You can use the CICS IA Affinities Reporter to produce files of basic transaction affinity groups for input to the Builder. The input to the Reporter, in the Affinity database objects, can be from several runs of the Collector, for example, against a production CICS region and a test CICS region, but must be for the same workload.

The rest of this section contains the following information:

- “Editing the CIUAFFBL job”
- “Syntax for input to the Builder” on page 160
- “Output from the Builder” on page 163

Editing the CIUAFFBL job

To run the Builder, edit and run the CIUAFFBL job.

Before running the CIUAFFBL job, change the values of the parameters listed below, as appropriate.

The changes you make to the job stream are similar to those described in Updating the Affinity database objects and are listed in the header of the JCL file.

- The JOB accounting parameters
- The PARM parameter of the EXEC statement

For example:

```
//BUILD EXEC PGM=CIUBLD,  
// PARM=('STATE=ACTIVE,MATCH=LUNAME,DSPSIZE=16',  
// 'CONTEXT=CICPLEX1')
```

[DSPSIZE={16|number}].

Specify the size, in the range 2 through 2000 (MB), of the data space created internally by the Builder to store the group tables.

[MATCH={LUNAME|USERID}].

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

[STATE={ACTIVE|DORMANT}].

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

[CONTEXT=plexname].

Specify the name, one through eight characters, of a CICSplex. If you specify

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

For more information about defining transaction groups to CICSplex SM, see *CICSplex SM Managing Business Applications*.

- The STEPLIB DD statement; specify the name of the CICS IA load library in which you have installed the Builder program, CIUBLD.
- The REPGRPS DD statement; specify the concatenation of names of the sequential data sets containing the basic affinity transaction groups to be input to the Builder. The Builder reads the lines of the input data sets and checks them for syntax and logic errors. For information about the valid syntax, see “Syntax for input to the Builder.”
- The AFFGRPS DD statement; specify the name of the sequential data set to which the combined affinity transaction groups are to be sent. This data set is suitable for input to CICSplex SM.
- The SYSPRINT DD statement; specify the destination for the report output by the Builder.

Syntax for input to the Builder

The syntax in the sequential data sets used as input for the Builder is similar, but not identical, to that allowed by CICSplex SM.

For more information, see *CICS Transaction Server for z/OS Installation Guide*. The differences are given in the following list:

- The only statements you can supply are:
 - CREATE statements for TRANGRPs and DTRINGRPs.
 - REMOVE statements for TRANGRPs.
 - TEXT statements and line comments. A line comment is a line that starts with an asterisk (*) in column 1.
 - HEADER statements for the Builder, but not for a CICSplex SM statement.
- Block comments delimited by '/*' and '*/' are not recognized.
- Transaction group names of up to 11 characters are allowed. CICSplex SM allows only 8 characters.
- A CREATE TRANGRP statement must have exactly one NAME, one AFFINITY, and one AFFLIFE value. MATCH and STATE values are optional and ignored; they are overridden by the values on the PARM statement or the default. A DESC value is optional and ignored. Any other keywords are reported as errors.
- A CREATE DTRINGRP statement must have exactly one TRANGRP and one TRANID value. Any other keywords are reported as errors.
- REMOVE TRANGRP statements are optional and are ignored by the Builder. However, if a REMOVE TRANGRP statement appears in an input data set, it must have exactly one NAME value. Any other keywords are reported as errors.
- CONTEXT statements in the input data set are optional and are ignored by the Builder. They are overridden by the CONTEXT operand of the PARM statement, if specified, or the default.
- A HEADER statement requires no keyword. APPLID, SAVEDATE, and SAVETIME are all optional, and, if specified, their values are not validated. The

HEADER statement must end in a semi-colon (;) and not span lines. Each input data set must start with a HEADER statement. See “HEADER statements” on page 162.

- If a line comment contains the characters HEADER anywhere in it, it is not treated as a comment and is parsed like any ordinary line in case it is a HEADER statement. Otherwise comment lines are thrown away.
- The only valid values for AFFINITY are GLOBAL, LUNAME, USERID, BAPPL , and LINK3270. NONE is not allowed.
- Keywords and values, including surrounding brackets, must not be split across input lines.
- Nested brackets are not allowed within values.
- The Builder is case-sensitive, both for keywords and their values. Keywords must be in uppercase.

Any syntax error causes an error message to be issued. Logic errors are also possible; for example, CREATE DTRINGRP before CREATE TRANGRP can cause error messages to be issued.

Any such errors do not cause the Builder to terminate immediately, but normally cause a skip to either the next keyword or the next statement, depending on the error. The Builder terminates with a return code of 8 when EOF is finally reached. An error report lists all errors encountered. For each error, the line containing the error is produced, with up to four preceding lines for the same statement to put the error in context, and the error message. The input syntax is shown in Figure 48 on page 162.

```

input_statement = {create_statement |
                  remove_statement |
                  header_statement |
                  context_statement |
                  comment}
create_statement = CREATE
                  {create_trangrp |
                  create_dtringrp}
;
create_trangrp = TRANGRP
                NAME (Trangroup)
                AFFINITY ({GLOBAL|LUNAME|USERID})
                AFFLIFE ({PERMANENT|SYSTEM|LOGON|SIGNON|PCONV})
                [DESC (string)]
                [MATCH ({LUNAME|USERID})]
                [STATE ({ACTIVE|DORMANT})]
create_dtringrp = DTRINGRP
                TRANGRP (Trangroup)
                TRANID (tranid)
remove_statement = REMOVE
                TRANGRP
                NAME (Trangroup)
;
context_statement = CONTEXT
                  [plexname]
;
header_statement = HEADER
                  [APPLID (applid)]
                  [SAVEDATE (date)]
                  [SAVETIME (time)]
;
comment = '*'
         [string |
         header_statement]

```

Figure 48. Builder input syntax

HEADER statements

The HEADER statement is specific to the Builder and is not a CICSplex SM statement. It is produced by the Affinities Reporter and is needed by the Builder to create unique transaction-group names.

The Affinities Reporter generates temporary transaction group names, for example, CW.00000001 and TS.00000001, while it is running, and stores these names in the output data set for that run. However, the Builder can take several Reporter data sets as input, and might therefore obtain the same transaction group name from different input data sets, describing different affinity transaction groups.

To ensure that the transaction group names are unique, the input transaction group names are qualified by the input data set name. To do this, when the Builder reads a HEADER statement, the first line of an input data set, it obtains the data set name from MVS. The HEADER statement is vital because without it the Builder cannot detect the change from one input data set to another.

If you omit a HEADER statement, the Builder might generate error messages or add transactions to the wrong group, and give incorrect line numbers in the error report and an incomplete report of data sets processed.

Output from the Builder

The Builder produces a file containing a set of definitions of combined affinity transaction groups and a report listing the combinations that occurred.

Combined affinity-transaction-group definitions

Before each definition of a combined group in the output file, the Builder adds a commented-out REMOVE command for that group. If you already have combined groups of the same name, check that it is appropriate to delete them before you uncomment the REMOVE command.

The name of each combined affinity transaction group is derived from the first alphanumeric transaction identifier in the combined group. For example, if ABCD was first, the transaction group name would be ABCDGRP.

For CICSplex SM, the name of each combined affinity transaction group must be unique.

Figure 49 shows a set of combined definitions. A MATCH filter of LUNAME, a STATE of ACTIVE, and a CONTEXT of CICPLEX1 were specified on the PARM parameter of the EXEC statement.

```
* HEADER APPLID(BUILDER ) SAVEDATE(07/09/27) SAVETIME(12:00:51); 1
*
* Generated by the CICS IA TRANSACTION AFFINITIES (Builder) on 2007/09/27
* Note: Suitable for input to CICSplex SM
*
CONTEXT CICPLEX1;
*
* REMOVE TRANGRP NAME(AFF1GRP );
CREATE TRANGRP NAME(AFF1GRP ) AFFINITY(LUNAME) AFFLIFE(SYSTEM )
    MATCH(LUNAME) STATE(DORMANT);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF1);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF2);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF3);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF4);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF5);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF6);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF7);
    CREATE DTRINGRP TRANGRP(AFF1GRP ) TRANID(AFF8);
*
* REMOVE TRANGRP NAME(AFTDGRP );
CREATE TRANGRP NAME(AFTDGRP ) AFFINITY(LUNAME) AFFLIFE(PCONV )
    MATCH(LUNAME) STATE(DORMANT);
    CREATE DTRINGRP TRANGRP(AFTDGRP ) TRANID(AFTD);
    CREATE DTRINGRP TRANGRP(AFTDGRP ) TRANID(AFTR);
    CREATE DTRINGRP TRANGRP(AFTDGRP ) TRANID(AFTW);
*
* REMOVE TRANGRP NAME(AUXXGRP );
CREATE TRANGRP NAME(AUXXGRP ) AFFINITY(GLOBAL) AFFLIFE(SYSTEM )
    MATCH(LUNAME) STATE(DORMANT);
    CREATE DTRINGRP TRANGRP(AUXXGRP ) TRANID(AUXX);
    CREATE DTRINGRP TRANGRP(AUXXGRP ) TRANID(CWA1);
```

Figure 49. Sample definitions for combined affinity transaction groups

Note:

1. The values of the SAVEDATE and SAVETIME fields in the HEADER statement give the latest save date and save time from any of the input data sets. See Figure 49 (**1**) and Figure 50 on page 166.

2. The combined transaction groups can be provided again to the Builder. For example, you might decide to:
 - a. Use the Affinities Reporter, then the Builder, to produce combined groups for temporary storage affinities.
 - b. Use the Affinities Reporter, then the Builder, to produce combined groups for all other affinity command types.
 - c. Merge the two files produced by the Builder in steps 2a and 2b, by providing those files to the Builder together.
 - d. Provide to CICSPlex SM the file produced by the Builder in step 2c.

Combining basic affinity transaction groups

When the Builder combines two basic affinity transaction groups, it assigns relations and lifetimes to the combined group based on the relations and lifetimes derived from the basic groups.

This assignment might cause some worsening of the relations and lifetimes. For example, LUNAME combined with USERID gives GLOBAL. Table 21 through Table 26 on page 165 show the relations and lifetimes that result from combining basic affinity transaction groups.

To help you analyze the effect of combining basic transaction affinity groups, the Builder produces a report that lists the combinations that occurred.

Table 21. Resulting affinity relations

Relation A	Relation B	Resulting relation C
GLOBAL	Any relation	GLOBAL
BAPPL	BAPPL	BAPPL
BAPPL	LUNAME	GLOBAL
BAPPL	USERID	GLOBAL
LINK3270	LINK3270	LINK3270
LINK3270	BAPPL	GLOBAL
LINK3270	LUNAME	GLOBAL
LINK3270	USERID	GLOBAL
LUNAME	LUNAME	LUNAME
LUNAME	USERID	GLOBAL
USERID	USERID	USERID

Table 22. Resulting affinity lifetimes (LUNAME relation)

Lifetime X	Lifetime Y	Resulting lifetime Z
PERMANENT	Any lifetime	PERMANENT
SYSTEM	SYSTEM	SYSTEM
SYSTEM	LOGON	SYSTEM
SYSTEM	PCONV	SYSTEM
LOGON	LOGON	LOGON
LOGON	PCONV	LOGON
PCONV	PCONV	PCONV

Table 23. Resulting affinity lifetimes (BAPPL relation)

Lifetime X	Lifetime Y	Resulting lifetime Z
PERMANENT	Any lifetime	PERMANENT
SYSTEM	Any other combination	SYSTEM
PROCESS	PROCESS	PROCESS
PROCESS	ACTIVITY	SYSTEM
ACTIVITY	PROCESS	SYSTEM
ACTIVITY	ACTIVITY	ACTIVITY

Table 24. Resulting affinity lifetimes (USERID relation)

Lifetime X	Lifetime Y	Resulting lifetime Z
PERMANENT	Any lifetime	PERMANENT
SYSTEM	SYSTEM	SYSTEM
SYSTEM	SIGNON	SYSTEM
SYSTEM	PCONV	SYSTEM
SIGNON	SIGNON	SIGNON
SIGNON	PCONV	SIGNON
PCONV	PCONV	PCONV

Table 25. Resulting affinity lifetimes (LINK3270 relation)

Lifetime X	Lifetime Y	Resulting lifetime Z
PERMANENT	Any lifetime	PERMANENT
SYSTEM	SYSTEM	SYSTEM
SYSTEM	FACILITY	SYSTEM
FACILITY	FACILITY	FACILITY

Table 26. Resulting affinity lifetimes (GLOBAL relation)

Lifetime X	Lifetime Y	Resulting lifetime Z
PERMANENT	Any lifetime	PERMANENT
Any other lifetime combination		SYSTEM

Data sets processed report

The data sets processed report gives the names of all the input data sets, specified on the REPGRPS DD statement, that were read.

This report is produced even if errors occur in the input data sets.

Only data sets that contain a HEADER statement appear in the report.

Dataset Name	CICS APPLID	Collector Last Save Date	Collector Last Save Time
CICSPDN1.TRANGRPS.MERGE1	CICSPDN1	07/10/25	09:05:09
CICSPDN1.TRANGRPS.MERGE2	CICSPDN1	07/10/26	15:22:34
CICSPDN1.TRANGRPS.ONE	CICSPDN1	07/10/27	12:00:51

Figure 50. Sample data sets processed report

Empty transaction groups report

The empty transaction groups report gives all basic transaction groups (Trangroups) that were defined, but contained no transactions.

It is produced only if the input data sets have no errors. An empty Trangroup probably indicates that you have made a mistake. The Reporter cannot produce empty Trangroups, so you must have created the input by hand, and probably omitted some corresponding CREATE DTRINGRP statements.

CICSPDN1.TRANGRPS.EMPTY1				
G1 (GLOBAL SYSTEM)	G2	(GLOBAL PERMANENT)	G3	(GLOBAL SYSTEM)
CICSPDN1.TRANGRPS.EMPTY2				
L2 (LUNAME PERMANENT)	L3	(LUNAME LOGON)	L4	(LUNAME PCONV)

Figure 51. Example empty Trangroups report

Group merge report

For each combined group, the group merge report gives the constituent transactions and basic groups that comprise the combined group and provides a type of audit trail.

It is produced only if there are no errors in the input data sets. It is very useful when establishing which basic group has caused the severe worsening of an affinity lifetime. For example, in Figure 52 on page 167, four groups were merged: three were LUNAME and PCONV, and one was LUNAME and SYSTEM. The latter caused the lifetime worsening.

```
Trangroup   : AFF1GRP
Affinity    : LUNAME
Lifetime    : SYSTEM
Match       : LUNAME
State       : DORMANT
  Consists of Transactions
    AFF1 AFF2 AFF3 AFF4 AFF5 AFF6 AFF7 AFF8
  Consists of groups merged from
    CICSPDN1.TRANGRPS.MERGE1
      TS.00000001 (LUNAME PCONV )   TS.00000002 (LUNAME PCONV )
    CICSPDN1.TRANGRPS.MERGE2
      TS.00000001 (LUNAME SYSTEM )   TS.00000002 (LUNAME PCONV )
Trangroup   : AFTDGRP
Affinity    : LUNAME
Lifetime    : PCONV
Match       : LUNAME
State       : DORMANT
  Consists of Transactions
    AFTD AFTR AFTW
  Consists of groups merged from
    CICSPDN1.TRANGRPS.ONE
      TS.00000001 (LUNAME PCONV )   TS.00000002 (LUNAME PCONV )
Trangroup   : AUXXGRP
Affinity    : GLOBAL
Lifetime    : SYSTEM
Match       : LUNAME
State       : DORMANT
  Consists of Transactions
    AUXX CWA1
  Consists of groups merged from
    CICSPDN1.TRANGRPS.ONE
      CW.00000001 (GLOBAL SYSTEM )
```

Figure 52. Sample group merge report

Error report

The error report gives the syntax or logic of any errors that were detected in the processing of the input files.

BUILDER REPGRPS ERROR REPORT

Dataset = CICS PDN1.TRANGRPS.ERR1

Line Number Statement in error

```
-----  
5 CREATE TRANGRP NAME(G3          ) AFFINITY(GLOBAL) AFFLIFE(LOGON  );  
  CIUAU5038 INVALID AFFLIFE for AFFINITY.  
6 CREATE TRANGRP NAME(G4          ) AFFINITY(GLOBAL) AFFLIFE(SIGNON );  
  CIUAU5038 INVALID AFFLIFE for AFFINITY.  
7 CREATE TRANGRP NAME(G5          ) AFFINITY(GLOBAL) AFFLIFE(PCONV );  
  CIUAU5038 INVALID AFFLIFE for AFFINITY.
```

Dataset = CICS PDN1.TRANGRPS.ERR2

Line Number Statement in error

```
-----  
11 CREATE TRANGRP NAME(L4          ) AFFINITY(LUNAME) AFFLIFE(SIGNON );  
  CIUAU5038 INVALID AFFLIFE for AFFINITY.  
15 CREATE TRANGRP NAME(U3          )  
16      AFFINITY(USERID) AFFLIFE(LOGON  );  
  CIUAU5038 INVALID AFFLIFE for AFFINITY.
```

Figure 53. Sample error report

Each error is accompanied by a message. For a description of the message, see Appendix D, “Messages and codes,” on page 255.

Chapter 13. Running the sample DB2 query

This section describes how to run the sample CICS IA batch query job CIUJSAMP.

It also describes how the supplied samples are run using the DB2 SQL Processing Using File Input (SPUFI) interface. The sample job uses the DB2 program DSNTEP2. It is a sample program and must be compiled, link-edited, and bound as usual. These programs are documented in the *DB2 Utility Guide and Reference* and the *DB2 Application Programming and SQL Guide*. SPUFI is a part of the distributed DB2 product. An installation job is used to bind it. For further information, see the *DB2 Utility Guide and Reference*.

The CIUJSAMP job

This CIUJSAMP job produces SQL output for the sample query you have selected.

Run the sample query JCL by editing and running the job CIUJSAMP. Before running CIUJSAMP, edit it to meet the requirements of your system.

Tailoring the job for your environment

Steps to take to ensure that the CIUJSAMP job will run in your environment.

About this task

1. Edit the job card to meet the requirements of your own system.
2. Change the following parameters:
 - _dbid_**
Your DB2 identifier (ID)
 - _hlq_** The high-level qualifier for CICS IA
 - _db2hlq_**
The high-level qualifier of the DB2 SDSNLOAD data set
 - _db2runhlq_**
The high level qualifier for the DB2 RUNLIB.LOAD data set
3. Select the SCIUSQL sample you want to run. Each sample contains several SQL queries. You can review and edit these members before running the job.

Running SPUFI

The sample SQL members can be run using the DB2 supplied utility SPUFI.

Contact your DB2 administrator to see if SPUFI is available at your site.

Chapter 14. Solving problems

This section helps you to isolate and determine the cause of CICS IA problems.

It contains these sections:

- “Overview of CICS IA problem determination”
- “Dealing with errors”
- “Contacting IBM Support” on page 176

Overview of CICS IA problem determination

This section describes tools and techniques that you can use to find the cause of a problem, and suggests one or more actions for solving it.

Software problems are generally defined by a symptom or set of symptoms. Problem determination involves classifying the symptoms and tracing them back to the source of the error.

Sometimes you cannot solve the problem yourself. For example, limitations in the hardware or software you are using could be causing the problem. If the cause of the problem is in the CICS IA code, you might need to contact IBM for further help.

CICS IA interfaces with several other components:

- DB2
- One or more CICS address spaces
- VSAM RLS
- The z/OS operating system
- The CICS IA plug-in for CICS Explorer

Problem determination might involve examining information from other products. If you determine that the error lies outside the CICS IA components reference the problem determination documentation for these other products.

Dealing with errors

Steps to take to try to solve errors with CICS IA.

Preliminary checks

Before looking further for the cause of the problem, review the following preliminary checks. These checks might highlight a simple cause or, at least, narrow the range of possible causes.

About this task

As you go through the questions, make a note of anything that might be relevant to the problem. Even if the observations you record do not at first suggest a cause, they could be useful later if you need to carry out systematic problem determination.

1. Has CICS IA executed successfully before?

If CICS IA has not run successfully before perhaps an error has occurred during installation.

Have you applied all the prerequisite APARs?

Have you reviewed all the information in the CICS IA *Program Directory* and the latest information in the Preventive Service Planning (PSP) upgrade for CICS IA Version 3.2?

2. Are there any messages explaining the failure?

If a problem is encountered, the CICS IA runtime issues messages to the CINT transient data destination.

Check the CICS MSGUSR and JESMSGLG logs for any security-related messages: for example, message DFHXS1111 or ICH408I. Insufficient authority to start required CICS IA transactions can result in various failure symptoms.

3. Can you reproduce the error?

If the failure can be reproduced, note the exact sequence of events required to recreate the problem. IBM support personnel might request the activation of various traces to determine the cause of the error situation.

4. What to do next

If the preliminary checks have enabled you to find the cause of the problem, you can now resolve it. Use other information in the CICS library and in the libraries of other licensed programs. If you have not yet found the cause, look at the problem in greater detail. Begin by trying to classify the type of problem.

Classifying the problem

One of the first requirements in solving a CICS IA problem is to determine what type of problem you are experiencing.

Problems can be classified into the following areas:

- Incorrect output or unexpected results
- Waits or hangs
- High CPU usage, perhaps caused by looping

Problems involving incorrect output or unexpected results can sometimes be the most difficult problems to solve. If you can re-create the problem, activating various levels of tracing usually provides IBM support personnel with sufficient information to solve the problem.

If the problem is a wait or hang, or high CPU usage on the CICS IA server, IBM support personnel might require a dump of the CICS IA address space.

Supplying a CICS IA trace

For certain problem types, you might be asked to supply a CICS IA trace. To switch tracing on and off, use the Global Options Menu.

The Global Options Menu, CIU300, is shown in Figure 35 on page 114.

CICS IA trace is written to an internal trace table and is visible only from within a dump of the CICS IA address space, as explained in the next section.

Taking a dump of CICS IA

For certain types of problem, you might be asked to supply a dump of CICS IA. To request a dump of a CICS IA address space and its associated data space, use the MVS DUMP command.

You can issue the MVS DUMP command from the console. 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 a five-character number that can be obtained from the CICS IA Operations Statistics Menu, CIU150, shown in Figure 54. It is usually 00000INT.

```

CIU150          CICS Interdependency Analyzer for z/OS - V3R2M0      2011/01/26
                  Statistics Menu for                               12:18:04PM

          CICS Sysid   : TS0A      CICS Applid   : IYCYZC3A

CINT state . . . . . : RUNNING      Collecting Dependencies

Records written last save. : 0
Total records on file. . . : 898
Total command flow records. : 0

Date/time of last start. . : 2011/01/25 02:58:29PM
Date/time of last save . . :
Date/time of last change . : 2011/01/25 02:58:32PM

Total time RUNNING . . . . : 0000:00:03 (HHHH:MM:SS)
Total time PAUSED. . . . . : (HHHH:MM:SS)

Table dataspace name . . . : 00000INT      0.9 % full

CICS Sysid: TS0A   CICS Applid: IYCLZC3A   TermID: TC69

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

```

Figure 54. Collector Statistics Menu panel, CIU150, showing the CICS IA data space name

To obtain the data space name for the CICS address space in which CICS IA is active, use the MVS DISPLAY ACTIVE command:

```
/D A,cicsname
```

where *cicsname* is the name of the CICS address space.

Figure 55 on page 174 shows some example output from the MVS DISPLAY ACTIVE command. In this example, the *cicsname* is CICSTS0A, the ASID is 00A8 and the data space name is 00000INT. The data space name is also available from the CICS IA Operations Statistics Menu, shown above. In the following examples a dump for CICS region CICSTS0A has been taken, where CICSTS0A is the name of the CICS address space. The CICS applid for this region is IYCYZC3A and the CICS sysid is TS0A.

```

RESPONSE=MV2E
IEE115I 14.59.54 2011.309 ACTIVITY 001
  JOBS      M/S    TS USERS   SYSAS   INITS   ACTIVE/MAX VTAM   OAS
00006      00089   00006   00034   00041   00006/00090     00053
  CICSTS0A CICSTS0A CICS      NSW SO  A=00A8 PER=NO SMC=000
                                PGN=N/A DMN=N/A AFF=NONE
                                CT=005.849S ET=01.55.31
                                WUID=STC00974 USERID=CT00LUSR
                                WKL=GENERAL SCL=STCUSER P=1
                                RGP=N/A SRVR=YES QSC=NO
                                ADDR SPACE ASTE=04937A00
                                DSPNAME=DFHDT001 ASTE=5EA66400
                                DSPNAME=00000INT ASTE=049CCC80

```

Figure 55. Example output from an MVS DISPLAY ACTIVE command, showing a data space name of 00000INT

To dump this data space, use the following command, where *nn* and *mmm* are the outstanding reply numbers:

```

/DUMP COMM=(IA DUMP)
/R nn,JOBNAME=CICSTS0A,CONT
/R mmm,DSPNAME=('CICSTS0A'.00000INT),END

```

Formatting the CICS IA Dump

If you use the MVS interactive problem control system (IPCS) to format and analyze CICS system dumps, you can use the CICS IA system dump formatting routine to format the CICS IA Collector data areas and trace, the dump formatting routine accessible to IPCS, as described in “CICS IA supplied modules required in the MVS linklist” on page 66. When your CICS system dump is formatted by the CICS release-dependent formatting routine, the CICS IA Collector data areas and trace are also formatted under the heading “CICS Interdependency Analyzer Control Blocks”.

Use the IPCS command:

```
IPCS VERBX DFHPDnnn 'ft'
```

where *nnn* is the CICS TS version; for example DFHPD650. This command invokes the feature table and routine for a CICS TS dump. The CICS IA feature routines are called CIUIADUF and CIUICDUF, and can be located in the output by searching on **=CIUIADUF** (or **=CIUICDUF**).

For example:

```

=CIUIADUF: Feature routine
===IA: CICS Interdependency Analyser Control Blocks
Collector status: FAILING           Collecting: INTERDEPENDENCIES
Date last started: 20110125        Time last started: 145212
Date last saved:                   Time last saved:
Saves: 0      Records last save: 0  Pauses: 0      Records total: 146

```

CICS IA plug-in for CICS Explorer level

The CICS IA plug-in and APAR level are obtained from two places. The IBM Support team will ask for one or both of these.

About this task

- The level of the SMP/E zip file in SCIUJAVE or SCIUJAVK can be obtained by querying the CIU_VERSION table:
SELECT DB_APAR_LEVEL,EXP_APAR_LEVEL,EXP_MIN_VER,EXP_LATEST_VER FROM CIU_VERSION

where:

DB_APAR_LEVEL

The APAR level of the latest changes to the CICS IA database.

EXP_APAR_LEVEL

The APAR level of the latest changes to the CICS IA plug-in.

EXP_MIN_VER

The minimum level of the CICS IA plug-in required on your workstation.

EXP_LATEST_VER

The latest level of the CICS IA plug-in that is available to download.

The CICS IA plug-in startup routine checks to see if it is equal to, or greater than, the EXP_MIN_LEVEL and issues an appropriate message to inform you to download and install the latest level of the CICS IA plug-in. The startup routine also checks against the EXP_LATEST_VER. If the current version is earlier then you are informed that a later version of the CICS IA plug-in is available to download.

Obtaining an error log

How to gather required documentation.

About this task

To obtain an error log:

Procedure

1. Click **Help>About IBM CICS Explorer** to open the About IAExplorer window.
2. Click **Configuration Details**.
3. Click **View Error Log**.
4. Click an appropriate Web browser in the Open With window, for example, Notepad.
5. Click **File>Save As** to save the log to an appropriate directory.

Obtaining configuration details

How to gather required documentation.

About this task

To obtain configuration details:

Procedure

1. Click **Help>About IBM CICS Explorer** to open the About IAExplorer window.
2. Click **Configuration Details**.
3. Click **Copy to clipboard**.
4. Paste in an appropriate file.

Viewing Eclipse plug-ins

How to gather required documentation.

About this task

How to view Eclipse plug-ins, shipped with the CICS Explorer, and their levels:

Procedure

1. Click **Help>About IBM CICS Explore** to open the About CICS IA plug-in window.
2. Click **Plug-in Details**. The CICS IA plug-ins are provided by IBM. You might be required to check the version number of the plug-ins.

Contacting IBM Support

The IBM Support structure helps you to resolve problems with IBM products and ensure that you can make the best use of your IBM computing systems. Program support is available to all licensed users of IBM licensed programs.

You can obtain help with your IBM software in any of the following ways:

- By searching for a solution at the CICS Technical Support Page at <http://www-306.ibm.com/software/htp/cics/ianaly/support/>
At this site you can, for example:
 - View Technotes and FAQs
 - Open a problem electronically, using the Electronic Service Request (ESR) form
- If you are a registered IBMLink user, by logging on to IBMLink at: <http://www.ibm.link.ibm.com/>
- By telephone call to your local Support Center.

The following topics help you decide when to contact the Support Center, and what information you need to collect before contacting the Center.

When to contact the Support Center

Before contacting the Support Center, try to ensure that the problem cannot be resolved without IBM's assistance.

In practice, many of the problems reported to Software Support turn out to be user errors. Other reported problems either cannot be reproduced or need to be handled by other parts of IBM Service, such as Hardware Customer Engineering or Systems Engineering. If you have followed the suggestions in this book and investigated all possible causes without finding the answer to your problem it is time to contact the Support Center.

Working with the Support Center

When you call the Support Center, your first contact will be with a Support Center operator. The operator records some initial details about your problem, and then places it on a problem queue. You might be transferred directly to a technical support specialist or you might receive a call back from a Support Center representative.

The Support Center needs to know as much as possible about your problem. Have the following information ready before making your first call:

- Your organization's name
- Your IBM Support Services' access code
- The suspected source of the problem
- The severity of the problem
- A complete description of the problem

“Information data sheet” on page 177 is a checklist of the additional, problem-related, information.

Information data sheet

This data sheet lists the problem determination information you need to have available when you contact IBM Support Services.

Table 27. Data sheet of problem determination information for IBM Support Center

Information for CICS IA plug-in for CICS Explorer problems
CICS IA plug-in for CICS Explorer operating system level, including the service pack level if any.
CICS IA "Help About" service level.
Frequency of the failure.
CIU_VERSION table information.
Can you recreate the failure?
When did the failure first occur?
Has the failing function ever worked correctly?
Detailed information about the CICS IA plug-in navigation or screen shots of the navigation.
CICS IA DB2 Host Connection Values.
CICS IA plug-in exception log files.
CICS IA plug-in popup messages.
Information on any recent maintenance that has been applied.
Information for CICS IA host problems
z/OS operating system level.
DB2 version and maintenance level.
Service levels for CICS and CICS IA.
The complete CICS region job logs, including CEEMSG and MSGUSR.
Information on any recent PTFs that have been applied.
CICS region auxiliary trace data sets.
Any dumps or other diagnostic information which were produced.
Frequency of the failure.
Can you recreate the failure?
When did the failure first occur?
Has the failing function ever worked correctly?

Appendix A. Details of dependencies and affinities collected

This section describes:

- The programming commands that are detected by the CICS IA Collector and what is reported for each type of command.
- The differences, if any, between what the Collector detects and what the Load Module Scanner detects. In general, the Load Module Scanner always detects more, because it covers paths that might not get exercised by the Collector, and because it cannot see beyond the command argument 0 to eliminate EXEC CICS commands that do not cause dependencies or affinities.

This information adds detail to the general description of “What can be monitored” on page 17.

Note:

1. In all cases, the Load Module Scanner reports only the verbs detected, without the resource name or, where appropriate, the SYSID.
2. For some commands, the Collector does not store the resource name. For example, on an EXEC CICS FEPI CONVERSE DATASTREAM CONVID command, when the dependency is between a FEPI program and a FEPI conversation, the Collector does not store the ID of the conversation, because of its transitory nature. When the resource name is not stored, the command is not saved in the DB2 tables in the Dependency database objects. The command is, however, reported by the Reporter.

This section contains these topics:

- “Commands monitored for potential dependencies”
- “Commands monitored for potential affinities” on page 192

Commands monitored for potential dependencies

This section lists the commands monitored by the Collector because they have the potential to create dependencies.

- “CICS commands detected”
- “CICS FEPI commands detected” on page 190
- “Non-CICS API commands detected” on page 191

CICS commands detected

The CICS commands detected by the Collector are listed here, excluding Front End Programming Interface (FEPI) commands, which are listed in “CICS FEPI commands detected” on page 190.

CICS API commands

This section lists the presentation commands.

Presentation commands

Atom Service command:

Command detected:

- EXEC CICS BIF DIGEST RECORD

BMS commands:

Commands detected:

- EXEC CICS PURGE MAP
- EXEC CICS RECEIVE MAP
 - Primary resource captured: Map Name
 - Secondary resource captured (if specified): Mapset Name
- EXEC CICS ROUTE
- EXEC CICS SEND MAP
 - Primary resource captured: Map Name
 - Secondary resource captured (if specified): Mapset Name
- EXEC CICS SEND TEXT

The dependency here is between a program and a mapset. Where only the map name is specified BMS uses the map name as the mapset name.

Terminal control and information commands:

Commands detected:

- EXEC CICS ADDRESS TCTUA
- EXEC CICS ASSIGN ALTSCRNHT
- EXEC CICS ASSIGN ALTSCRNWD
- EXEC CICS ASSIGN APLKYBD
- EXEC CICS ASSIGN APLTEXT
- EXEC CICS ASSIGN COLOR
- EXEC CICS ASSIGN DEFSCRNHT
- EXEC CICS ASSIGN DEFSCRNWD
- EXEC CICS ASSIGN DS3270
- EXEC CICS ASSIGN EXTDS
- EXEC CICS ASSIGN HIGHLIGHT
- EXEC CICS ASSIGN MAPCOLUMN
- EXEC CICS ASSIGN MAPHEIGHT
- EXEC CICS ASSIGN MAPLINE
- EXEC CICS ASSIGN MAPWIDTH
- EXEC CICS ASSIGN PS
- EXEC CICS ASSIGN SCRNHT
- EXEC CICS ASSIGN SCRNWD
- EXEC CICS ASSIGN SYSID
- EXEC CICS ASSIGN TEXTKYBD
- EXEC CICS ASSIGN TEXTPRINT
- EXEC CICS ISSUE COPY
- EXEC CICS ISSUE PASS
- EXEC CICS ISSUE RESET
- EXEC CICS SIGNOFF
- EXEC CICS SIGNON

The option specified on the EXEC CICS commands is reported as the resource.

APPC mapped conversation commands without the CONVID option:

Commands detected:

- EXEC CICS CONVERSE
- EXEC CICS EXTRACT PROCESS
- EXEC CICS ISSUE ABEND
- EXEC CICS ISSUE CONFIRMATION
- EXEC CICS ISSUE SIGNAL

- EXEC CICS ISSUE ERROR
- EXEC CICS ISSUE DISCONNECT
- EXEC CICS RECEIVE
- EXEC CICS SEND

Distributed transaction processing (DTP) commands:

Commands detected:

- EXEC CICS ALLOCATE
- EXEC CICS CONNECT PROCESS
- EXEC CICS CONVERSE CONVID
- EXEC CICS CONVERSE SESSION
- EXEC CICS SEND SESSION
- EXEC CICS FREE CONVID

This dependency is between a program and a remote transaction or process.

The convid or session returned on the ALLOCATE call is stored with the SYSID or session name in a temporary table. The ALLOCATE call is reported, with the SYSID or session as a resource.

Every CONNECT PROCESS, SEND SESSION, CONVERSE CONVID, CONVERSE SESSION, or FREE CONVID is matched by convid against the table entries. If the CONVID and SESSION or CONVID or SESSION match a temporary table entry:

- For CONNECT PROCESS, the PROCNAME and previously specified SYSID/SESSION from the ALLOCATE are reported. The temporary table entry is deleted.
- For SEND SESSION, the first four characters of data are assumed to be the process name. This process name and the previously specified SYSID/SESSION are reported. If the command is successful, the temporary table entry is deleted, because the remote system and remote process name are now associated.
- CONVERSE CONVID, CONVERSE SESSION is the same as SEND SESSION, above.
- For FREE, the temporary table entry is deleted. No information is reported, because no process was started.

Event command:

Command detected:

- EXEC CICS SIGNAL EVENT

This dependency is between a program and business event processed.

The resource reported is the event.

Exception support:

Command detected:

- EXEC CICS HANDLE ABEND PROGRAM

This dependency is between a program and another local program.

The program name is reported as the resource.

File Control commands:

Commands detected:

- EXEC CICS DELETE FILE()
- EXEC CICS ENDBR FILE()
- EXEC CICS INQ NEXT FILE()
- EXEC CICS READ FILE()
- EXEC CICS READ UPD FILE()
- EXEC CICS READNEXT FILE()
- EXEC CICS READPREV FILE()
- EXEC CICS RESETBR FILE()
- EXEC CICS REWRITE FILE()
- EXEC CICS STARTBR FILE()
- EXEC CICS UNLOCK FILE()
- EXEC CICS WRITE FILE()

This dependency is between a program and a file, possibly remote.

The resource reported is the file name.

If the command is shipped to a remote region, the system identifier (SYSID) of the remote region is reported.

Interval Control commands:

Command detected:

- EXEC CICS START
- EXEC CICS RETURN

This dependency is between a program and a transaction, local or remote.

The resource reported is the transaction name.

If the command is shipped or routed to a remote region, the SYSID of the remote region is reported.

If the CHANNEL option is specified on the command the channel name is also reported as a resource.

These resources are captured when the "Transactions" flag for transaction CINT, panel CIU240, is set to Y or D.

Journal Control commands:

Commands detected:

- EXEC CICS WAIT JOURNALNAME
- EXEC CICS WAIT JOURNALNUM
- EXEC CICS WRITE JOURNALNAME
- EXEC CICS WRITE JOURNALNUM

This dependency is between a program and a CICS journal.

The resource reported is the journal number.

Named Counter Server commands:

Commands detected:

- EXEC CICS DEFINE COUNTER and DEFINE DCOUNTER

- EXEC CICS DELETE COUNTER and DELETE DCOUNTER
- EXEC CICS GET COUNTER and GET DCOUNTER
- EXEC CICS QUERY COUNTER and QUERY DCOUNTER
- EXEC CICS REWIND COUNTER and REWIND DCOUNTER
- EXEC CICS UPDATE COUNTER and UPDATE DCOUNTER

This dependency is between a program and a named counter in a named counter pool in the coupling facility.

The resource reported is the named counter.

If the POOL option is specified on the command there is an additional dependency between a program and a pool resource. Therefore the pool name is also reported as a dependency resource.

Program Control commands:

An alphabetic list of commands detected:

- EXEC CICS LOAD
 - This dependency is between a program and another local program.
 - The program name is reported as the resource.
- EXEC CICS LINK
 - This dependency is between a program and another program, possibly remote.
 - The program name is reported as the resource.
 - If the command is routed to a remote region, the SYSID of the remote region is reported.
 - If the CHANNEL option is specified on the command there is an additional dependency between the program and the channel resource. The channel name is reported as a resource.

EXEC CICS XCTL

- This dependency is between a program and another local program.
- The program name is reported as the resource.
- If the CHANNEL option is specified on the command there is an additional dependency between the program and the channel resource. The channel name is reported as a resource.
- EXEC CICS RETURN TRANSID
 - This dependency is between a program and a local or remote transaction.
 - The transaction name is reported as a resource.
 - If the CHANNEL option is specified on the command there is an additional dependency between the program and the channel resource. The channel name is reported as a resource.
- EXEC CICS GET CONTAINER
- EXEC CICS PUT CONTAINER
- EXEC CICS MOVE CONTAINER
- EXEC CICS DELETE CONTAINER
 - This dependency is between a program and a container.
 - The CONTAINER name is reported as the resource.

- If the CHANNEL option is specified on the command there is an additional dependency between the program and the channel resource. The channel name is reported as a resource.

Task control commands:

Commands detected:

- EXEC CICS ENQ
- EXEC CICS DEQ

This dependency is between a program and the resource that it is enqueued upon. The name of the resource, limited to the first 50 characters, is reported.

Temporary Storage commands:

Commands detected:

- EXEC CICS READQ TS
- EXEC CICS WRITEQ TS
- EXEC CICS DELETEQ TS

This dependency is between a program and a Temporary Storage queue, either local or remote.

The resource reported is the Temporary Storage queue name.

If the command is shipped to a remote region, the SYSID of the remote region is reported.

If the current task number is detected in the Temporary Storage queue name, it is shown as +TA+ in the resource name by the Reporter. Using +TA+ prevents all references to unique Temporary Storage queues being recorded, causing unnecessary filling of the dependency table and file.

If the current terminal identifier is detected in the Temporary Storage queue name, it is indicated by +TE+ in the Reporter. Using +TE+ prevents storage wastage in the dependency table and file.

Transient Data commands:

Commands detected:

- EXEC CICS READQ TD
- EXEC CICS WRITEQ TD
- EXEC CICS DELETEQ TD

This dependency is between a program and a Transient Data queue, either local or remote.

The resource reported is the Transient Data queue name.

If the command is shipped to a remote region, the SYSID of the remote region is reported.

WEB commands:

Commands detected:

- WEB ENDBROWSE

- WEB EXTRACT
- WEB READ
- WEB READNEXT
- WEB RECEIVE
- WEB RETRIEVE
- WEB SEND
- WEB OPEN
- WEB CLOSE
- WEB CONVERSE
- WEB PARSE
- WEB STARTBROWSE
- WEB WRITE

A resource is not recorded for WEB commands. The only fact that is recorded for a WEB command is that the WEB command has been issued.

Web Services Addressing commands:

Commands detected:

- EXEC CICS WSACONTEXT BUILD
- EXEC CICS WSACONTEXT GET
- EXEC CICS WSACONTEXT DELETE
- EXEC CICS WSAEPR CREATE

Exit commands:

Commands detected:

- EXEC CICS ENABLE PROGRAM
The dependency is between a program and an exit name. The exit name is reported as the resource.
- EXEC CICS DISABLE PROGRAM
The dependency is between a program and an exit name. The exit name is reported as the resource.
- EXEC CICS EXTRACT EXIT
The dependency is between a program and an exit name. The exit name is reported as the resource.
- CALL exit
The dependency is between the program and the called task related user exit. The exit name is reported as the resource.

Other commands:

Commands detected:

- EXEC CICS ADDRESS CWA
CWA is reported as the resource.
- EXEC CICS ASSIGN APPLID
APPLID is reported as the resource.
- EXEC CICS GETMAIN SHARED
If the ADDR option is specified on the command "ADDR" is reported as the resource.

XML parser commands:

Commands detected:

- EXEC CICS TRANSFORM XFORMTYPE(DATATOXML)
- EXEC CICS TRANSFORM XFORMTYPE(XMLTODATA)

CICS SPI commands

Commands detected:

File Control commands

- EXEC CICS CREATE FILE()
- EXEC CICS DISCARD FILE()
- EXEC CICS INQUIRE FILE()
- EXEC CICS SET FILE()

Atomservice commands:

Commands detected:

- EXEC CICS CREATE ATOMSERVICE
- EXEC CICS DISCARD ATOMSERVICE
- EXEC CICS INQUIRE ATOMSERVICE
- EXEC CICS INQUIRE ATOMSERVICE NEXT
- EXEC CICS SET ATOMSERVICE

INQUIRE BRFACILITY, SET BRFACILITY:

- This dependency is between a program and a local 3270 virtual terminal, bridge facility.
- The name of the bridge facility is reported as the resource.

Bundle commands:

Commands detected:

- EXEC CICS CREATE BUNDLE
- EXEC CICS DISCARD BUNDLE
- EXEC CICS INQUIRE BUNDLE
- EXEC CICS INQUIRE BUNDLE NEXT
- EXEC CICS SET BUNDLE
- EXEC CICS INQUIRE BUNDLEPART
- EXEC CICS INQUIRE BUNDLEPART NEXT

This dependency is between a program and application bundles.

CSD commands:

Commands detected:

- EXEC CICS CSD ADD GROUP
- EXEC CICS CSD ALTER RESTYPE
- EXEC CICS CSD APPEND LIST
- EXEC CICS CSD COPY GROUP
- EXEC CICS CSD COPY RESTYPE
- EXEC CICS CSD DEFINE RESTYPE
- EXEC CICS CSD DELETE LIST
- EXEC CICS CSD DELETE GROUP
- EXEC CICS CSD DELETE RESTYPE
- EXEC CICS CSD ENDBRGROUP
- EXEC CICS CSD ENDBRLIST
- EXEC CICS CSD ENDBRRSRCE

- EXEC CICS CSD GETNEXTGROUP GROUP
- EXEC CICS CSD GETNEXTLIST LIST
- EXEC CICS CSD GETNEXTRSRCE RESTYPE
- EXEC CICS CSD INQUIREGROUP GROUP
- EXEC CICS CSD INQUIREGROUP GROUP LIST
- EXEC CICS CSD INQUIRELIST LIST
- EXEC CICS CSD INQUIRERSRCE RESTYPE
- EXEC CICS CSD INSTALL LIST
- EXEC CICS CSD INSTALL GROUP
- EXEC CICS CSD INSTALL RESTYPE
- EXEC CICS CSD LOCK LIST
- EXEC CICS CSD LOCK GROUP
- EXEC CICS CSD REMOVE GROUP
- EXEC CICS CSD RENAME RESTYPE
- EXEC CICS CSD STARTBRGROUP
- EXEC CICS CSD STARTBRLIST
- EXEC CICS CSD STARTBRRSRCE
- EXEC CICS CSD UNLOCK LIST
- EXEC CICS CSD UNLOCK GROUP
- EXEC CICS CSD USERDEFINE RESTYPE
- EXEC CICS CSD DISCONNECT

CREATE CORBASERVER, INQUIRE CORBASERVER, SET CORBASERVER, PERFORM CORBASERVER, DISCARD CORBASERVER:

- This dependency is between a program and a local CorbaServer.
- The name of the CorbaServer is reported as the resource.

CREATE DB2ENTRY, INQUIRE DB2ENTRY, SET DB2ENTRY, DISCARD DB2ENTRY:

- This dependency is between a program and a local DB2ENTRY, used to define resources to be used by a specific transaction or group of transactions when accessing DB2.
- The name of the DB2ENTRY is reported as the resource.

CREATE DB2TRAN, INQUIRE DB2TRAN, SET DB2TRAN, DISCARD DB2TRAN:

- This dependency is between a program and a local DB2TRAN, associated with a DB2ENTRY.
- The name of the DB2TRAN is reported as the resource.

CREATE DJAR, INQUIRE DJAR, PERFORM DJAR, DISCARD DJAR:

- This dependency is between a program and a local deployed JAR file.
- The name of the deployed JAR file is reported as the resource.

Event commands:

Commands detected:

- EXEC CICS INQUIRE EVENTBINDING
- EXEC CICS INQUIRE EVENTBINDING NEXT
- EXEC CICS SET EVENTBINDING
- EXEC CICS INQUIRE EVENTPROCESS
- EXEC CICS SET EVENTPROCESS
- EXEC CICS INQUIRE CAPTURESPEC
- EXEC CICS INQUIRE CAPTURESPEC NEXT

This dependency is between a program and business event processed.

The resource reported is the events.

CREATE IPCONN, INQUIRE IPCONN, SET IPCONN, DISCARD IPCONN:

- This dependency is between a program and the name of the connection to the remote system or region.
- The connection name is reported as the resource.

CREATE FILE, INQUIRE FILE, SET FILE, DISCARD FILE:

- This dependency is between a program and a local file.
- The file name is reported as the resource.

JVM server commands:

Commands detected:

- EXEC CICS CREATE JVMSERVER
- EXEC CICS DISCARD JVMSERVER
- EXEC CICS INQUIRE JVMSERVER
- EXEC CICS INQUIRE JVMSERVER NEXT
- EXEC CICS SET JVMSERVER

INQUIRE JOURNALNAME, SET JOURNALNAME, DISCARD JOURNALNAME:

- This dependency is between a program and a CICS journal.
- The journal name is reported as the resource.

INQUIRE JOURNALNUM, SET JOURNALNUM, DISCARD JOURNALNUM:

- This dependency is between a program and a CICS journal.
- The journal number is reported as the resource.

INQUIRE JVMPROFILE:

- This dependency is between a program and a JVM profile.
- The name of the JVM profile is reported as the resource.

CREATE LIBRARY, INQUIRE LIBRARY, SET LIBRARY, DISCARD LIBRARY:

- This dependency is between a program and the library resource.
- The library name is reported as the resource.

MQ commands:

Commands detected:

- EXEC CICS CREATE MQCONN
- EXEC CICS INQUIRE MQCONN
- EXEC CICS DISCARD MQCONN
- EXEC CICS SET MQCONN
- EXEC CICS INQUIRE MQINI
- EXEC CICS DISCARD MQINI

CREATE PIPELINE, INQUIRE PIPELINE, SET PIPELINE, DISCARD PIPELINE, PERFORM PIPELINE:

- This dependency is between a program and a PIPELINE.
- The name of the PIPELINE is reported as the resource.

CREATE PROGRAM, INQUIRE PROGRAM, SET PROGRAM, DISCARD PROGRAM:

- This dependency is between a program and another local program.
- The program name is reported as the resource.

CREATE TCPIPSERVICE, INQUIRE TCPIPSERVICE, SET TCPIPSERVICE, DISCARD TCPIPSERVICE:

- This dependency is between a program and a local TCP/IP service.
- The name of the TCP/IP service is reported as the resource.

CREATE TDQUEUE, INQUIRE TDQUEUE, SET TDQUEUE:

- This dependency is between a program and a local transient data (TD) queue.
- The name of the TD queue is reported as the resource.

CREATE TRANSACTION, INQUIRE TRANSACTION, SET TRANSACTION, DISCARD TRANSACTION:

- This dependency is between a program and a local transaction.
- The transaction name is reported as the resource.

CREATE TSMODEL, INQUIRE TSMODEL, DISCARD TSMODEL:

- This dependency is between a program and a local temporary storage (TS) model.
- The name of the TS model is reported as the resource.

INQUIRE TSPPOOL:

- This dependency is between a program and a local shared temporary storage pool.
- The name of the TS pool is reported as the resource.

INQUIRE TSQNAME, SET TSQNAME:

- This dependency is between a program and a local temporary storage queue.
- The name of the TS queue is reported as the resource.

INQUIRE TSQUEUE, SET TSQUEUE:

- This dependency is between a program and a local temporary storage queue.
- The name of the TS queue is reported as the resource.

CREATE URIMAP, INQUIRE URIMAP, SET URIMAP, DISCARD URIMAP:

- This dependency is between a program and a URIMAP.
- The name of the URIMAP is reported as the resource.

CREATE WEBSERVICE, INQUIRE WEBSERVICE, SET WEBSERVICE, DISCARD WEBSERVICE:

- This dependency is between a program and a WEBSERVICE.
- The name of the WEBSERVICE is reported as the resource.

XML parser commands:

Commands detected:

- EXEC CICS INQUIRE XMLTRANSFORM
- EXEC CICS INQUIRE XMLTRANSFORM NEXT
- EXEC CICS SET XMLTRANSFORM

CICS FEPI commands detected

CICS FEPI API commands

An alphabetic list of the commands detected.

- FEPI ALLOCATE PASSCONVID
- FEPI CONVERSE DATASTREAM CONVID, FEPI CONVERSER FORMATTED CONVID
- FEPI EXTRACT CONV, FEPI EXTRACT FIELD, FEPI EXTRACT STSN
- FEPI FREE
- FEPI ISSUE
- FEPI RECEIVE DATASTREAM, FEPI SEND DATASTREAM
- FEPI RECEIVE FORMATTED, FEPI SEND FORMATTED
- FEPI REQUEST PASSTICKET

The dependency for these commands is between a FEPI program and a FEPI conversation. No resource name is reported. See note 2 on page 179.

- FEPI ALLOCATE POOL
- FEPI CONVERSE DATASTREAM POOL, FEPI CONVERSE FORMATTED POOL

This dependency for these commands is between a FEPI program and a FEPI pool. The name of the pool is reported as the resource.

- FEPI START

This dependency is between a FEPI program and a local CICS transaction or conversation. The name of the transaction, if available, is reported as the resource.

CICS FEPI SPI commands

Commands detected:

FEPI INQUIRE CONNECTION, FEPI SET CONNECTION:

- This dependency is between a FEPI program and a FEPI node and target.
- The name of the node is reported as the resource.

FEPI INQUIRE NODE, FEPI SET NODE:

- This dependency is between a FEPI program and a FEPI node.
- The name of the node is reported as the resource.

FEPI ADD POOL, FEPI INSTALL POOL, FEPI DELETE POOL, FEPI INQUIRE POOL, FEPI SET POOL, FEPI DISCARD POOL:

- This dependency is between a FEPI program and a FEPI pool.
- The name of the pool is reported as the resource.

FEPI INSTALL PROPERTYSET, FEPI INQUIRE PROPERTYSET, FEPI DISCARD PROPERTYSET:

- This dependency is between a FEPI program and a named set of FEPI properties.
- The name of the property set is reported as the resource.

FEPI INQUIRE TARGET, FEPI SET TARGET:

- This dependency is between a FEPI program and a FEPI target.

- The name of the target is reported as the resource.

Non-CICS API commands detected

DB2: EXEC SQL

Commands detected:

- ALTER
- CLOSE
- CREATE
- DELETE
- DESCRIBE
- DROP
- EXECUTE
- EXECUTE IMMEDIATE
- EXPLAIN
- FETCH
- INSERT
- OPEN
- PREPARE
- SELECT
- UPDATE

WebSphere MQ: MQM

Commands detected:

- MQCLOSE
- MQGET
- MQOPEN
- MQPUT
- MQPUT1
- MQBUFMH
- MQCB
- MQCTL
- MQCRTMH
- MQDLTMH
- MQDLTMP
- MQINQ
- MQINQMP
- MQMHBUF
- MQSETMP
- MQSTAT
- MQSUB
- MQSUBRQ

IMS database: EXEC DLI

Commands detected:

- DELETE
- GET NEXT
- GET NEXT IN PARENT
- GET UNIQUE
- INSERT
- REPLACE
- SCHEDULE

IMS database: CBLTDLI, ASMTDLI and PLITDLI calls

Commands detected:

- DELETE
- GET NEXT
- GET NEXT IN PARENT
- GET UNIQUE
- INSERT
- REPLACE
- SCHEDULE

Natural commands

Commands detected:

- CALL ADABAS
- CALL NATURAL PROGRAM

Commands monitored for potential affinities

This section lists the affinity-related EXEC CICS commands detected by the CICS IA Collector.

All commands listed here are *capable of* causing affinities; they might or might not actually do so.

CICS API commands

This section contains CICS API commands detected by the Collector, that might create *inter-transaction* affinities.

Interval Control commands

An alphabetic list of the commands detected.

- EXEC CICS CANCEL
- EXEC CICS DELAY
- EXEC CICS POST
- EXEC CICS RETRIEVE WAIT
- EXEC CICS START

Program Control commands

An alphabetic list of the commands detected.

- EXEC CICS LOAD
- EXEC CICS RELEASE

Storage Control commands

An alphabetic list of the commands detected.

- EXEC CICS FREEMAIN
- EXEC CICS GETMAIN SHARED

Task Control commands

An alphabetic list of the commands detected.

- EXEC CICS DEQ
- EXEC CICS ENQ

Temporary Storage commands

An alphabetic list of the commands detected.

- EXEC CICS DELETEQ TS
- EXEC CICS READQ TS

- EXEC CICS WRITEQ TS

Other commands

An alphabetic list of the commands detected.

- EXEC CICS ADDRESS CWA
- EXEC CICS COLLECT STATISTICS

The following CICS API commands, detected by the Collector, might create *transaction-system* affinities:

Business Transaction Services (BTS) commands

An alphabetic list of the commands detected.

- EXEC CICS ENDBROWSE ACTIVITY
- EXEC CICS ENDBROWSE CONTAINER
- EXEC CICS ENDBROWSE EVENT
- EXEC CICS ENDBROWSE PROCESS
- EXEC CICS GETNEXT ACTIVITY
- EXEC CICS GETNEXT CONTAINER
- EXEC CICS GETNEXT EVENT
- EXEC CICS GETNEXT PROCESS
- EXEC CICS STARTBROWSE ACTIVITY
- EXEC CICS STARTBROWSE CONTAINER
- EXEC CICS STARTBROWSE EVENT
- EXEC CICS STARTBROWSE PROCESS

CICS SPI commands

The following CICS SPI commands, detected by the Collector, might create *transaction-system* affinities.

- EXEC CICS CREATE
- EXEC CICS DISABLE PROGRAM
- EXEC CICS DISCARD
- EXEC CICS ENABLE PROGRAM
- EXEC CICS EXTRACT EXIT
- EXEC CICS INQUIRE
- EXEC CICS PERFORM
- EXEC CICS RESYNC
- EXEC CICS SET
- EXEC CICS WAITCICS
- EXEC CICS WAIT EVENT
- EXEC CICS WAIT EXTERNAL

Details of what is detected

This section describes what is detected by the Detector and Reporter for each affinity type.

Additionally, it highlights the differences, if any, with what the Scanner detects. (In general, the Scanner always detects more, because it covers paths that may not get exercised by the Detector, and because it cannot see beyond the command argument zero to eliminate commands that do not actually cause affinity.)

ENQ/DEQ

- The affinity here is between all transactions that ENQ or DEQ on a given resource. The match is made on the resource.

- It is possible for the ENQ/DEQ resource to be either a character string of length 1 to 255 bytes, or an address (which has an implied length of 4 bytes).
- The affinity relation can be GLOBAL, BAPPL, or USERID.
- Lifetime is always SYSTEM.
- Commands that result in a LENGERR condition are grouped together and treated as a resource of 'LENGERR'. Any other condition results in a valid resource and does not affect the treatment of the command.
- Because of affinity record size limitations, character string resources of greater than 207 bytes in length are compressed to 207 bytes. The compression is achieved by removing bytes from the middle of the string (these are probably less significant than those at either end). This means that such resources may be flagged as being the same when they are not, if the only variation is in the removed bytes. Check all such compressed resources to see if that is the case. The Reporter flags such compression, and pads the resource back out to the correct length for the report, by inserting '?' characters.

TS commands

- The affinity here is between all transactions that use the same TS queue. It applies to both MAIN and AUXILIARY TS. The match is made on the name of the TS queue.
- The affinity relation can be GLOBAL, BAPPL, LUNAME, or USERID.
- Lifetime can be PCONV, LOGON, SIGNON, ACTIVITY, PROCESS,SYSTEM, and PERMANENT. A MAIN queue cannot be recovered, regardless of definition, so cannot cause PERMANENT.
- No data is collected if a TS queue is defined as remote or if a remote SYSID is specified on the TS command. In such cases, the request is satisfied by a remote CICS region or by a temporary storage pool in the coupling facility.
- Commands in error are treated in the same way as commands that give a NORMAL response, so data is collected.
- If a TS queue is created and deleted within the same task, no data is collected.

Scanner differences: Scanner detects all instances of TS commands.

LOAD HOLD/RELEASE

- The affinity here is between all transactions that LOAD HOLD and RELEASE the same program (or, more probably, table). The match is made on the program name.
- The LOAD and RELEASE protocol applies only to programs that are defined with RELOAD(NO). If the Detector can not establish the RELOAD attribute for some reason, RELOAD(NO) is assumed.
- Once a LOAD HOLD has occurred for a program, any subsequent LOAD (with or without HOLD) or RELEASE is part of the affinity.
- The affinity relation is GLOBAL or BAPPL.
- Lifetime is always SYSTEM.
- Commands in error are treated in the same way as commands that give a NORMAL response, so data is collected.
- LOAD with no HOLD for programs defined as RESIDENT is not treated as an affinity because relying on residency for sharing is inherently unsafe, the program can be replaced by SET PROG() NEWCOPY.
- The incorrect use of RELEASE for a program defined with RELOAD(YES) is not detected.

Scanner differences: Scanner detects all instances of LOAD, not just LOAD HOLD, and all instances of RELEASE.

RETRIEVE WAIT/START

- The affinity here is between all the transactions that issue START commands for a particular transaction at a terminal, where that transaction issues RETRIEVE WAIT. The transaction that issues the RETRIEVE WAIT is also part of the affinity. The match is made on the transid.
- The affinity relation can be GLOBAL or USERID.
- Lifetime can be SYSTEM or PERMANENT. PERMANENT is assumed if PROTECT is specified on any START.
- If the transaction to be STARTed is defined as remote or a remote SYSID was specified on the START command so that the command is function shipped to a remote CICS region, no data is collected.
- Commands in error are treated in the same way as commands that give a NORMAL response, so data is collected.

Scanner differences: Scanner detects all instances of RETRIEVE WAIT, and all instances of START that either specify TERMID, or omit NOCHECK, or specify REQID (because of CANCEL affinity).

ADDRESS CWA

- The affinity here is between all transactions that issue ADDRESS CWA.
- The affinity relation is GLOBAL or BAPPL.
- Lifetime is always SYSTEM.

Scanner differences: None.

GETMAIN SHARED/FREEMAIN

- The affinity here is between the transaction that obtains storage via GETMAIN SHARED and the transaction that frees the same piece of storage via FREEMAIN. Both transactions must be seen for there to be affinity. The match is made on storage address.
- However, the situation is complicated by the fact that the storage address may be passed to other transactions; and if they access the storage, they cannot be detected, because the storage access does not take place through the CICS API.
- The affinity relation may be GLOBAL, BAPPL,LUNAME, or USERID.
- Lifetime can be PCONV, LOGON, SIGNON, ACTIVITY, PROCESS, or SYSTEM. However, the Detector always worsens LOGON and SIGNON to SYSTEM, because of limitations in the way that this affinity is detected.
- Commands in error are ignored, as there is no address for matching GETMAIN with FREEMAIN, no data is collected.
- A GETMAIN/FREEMAIN affinity is considered to be initiated from a terminal if the GETMAIN is initiated from a terminal. Whether the FREEMAIN was so initiated or not is irrelevant.
- Any unmatched GETMAIN SHAREDs are also reported if they have never matched by the time a Detector stop occurs. They are output in a separate report section. Note that on a start with restore data, they are not restored and are deleted from the affinity file.

Scanner differences: Scanner finds all instances of GETMAIN SHARED and all instances of FREEMAIN.

LOAD/FREEMAIN

- The affinity here is between the transaction that loads the program via LOAD and the transaction that releases the same program via FREEMAIN. The match is made on load point address.
- However, the situation is complicated by the fact that the load point address may be passed to other transactions (for example, the program is actually a table); and if they access the program, they cannot be detected. This is analogous to storage address passing with GETMAIN SHARED/FREEMAIN.
- The LOAD and FREEMAIN protocol applies only to programs defined as RELOAD(YES). Note that HOLD is irrelevant, as CICS Program Control never sees the FREEMAIN, or knows the storage location of the individual task's copy, and so cannot release the program at task end. This implies that all LOADs must be examined as they are all effectively LOAD HOLDs.
- The affinity relation may be GLOBAL, BAPPL, LUNAME, or USERID.
- Lifetime can be PCONV, LOGON, SIGNON, ACTIVITY, PROCESS, or SYSTEM. However, the Detector always worsens LOGON and SIGNON to SYSTEM, because of limitations in the way that this affinity is detected.
- Commands in error are ignored, because there is no load address on which to match LOAD with FREEMAIN, so no data is collected. LOADs with no SET option are ignored, because no load address is returned, so no data is collected.
- A LOAD/FREEMAIN affinity is considered to be initiated from a terminal if the LOAD is initiated from a terminal. Whether the FREEMAIN was so initiated or not is irrelevant.
- Any unmatched LOADs are also reported if they have never matched by the time a Detector stop occurs. They are output in a separate report section. Note that on a start with restore data, they are not restored and are deleted from the affinity file.

Scanner differences: Scanner finds all instances of LOAD and all instances of FREEMAIN.

CANCEL/DELAY/POST/START

- The affinity here is between the transaction that issues the DELAY, POST or START command and the transaction that issues the CANCEL command via REQID. The match is on REQID.
- In order for another task to CANCEL a DELAY, REQID must be explicitly specified on the DELAY command. If no REQID is specified on a DELAY command, it cannot be canceled, and therefore cannot be detected. In order for another task to CANCEL a START or POST, it is not necessary to specify REQID on the command because CICS supplies a unique REQID that may be used (unless START specifies NOCHECK). So only START commands that do not both specify NOCHECK and omit REQID, and all POST commands, are detected.
- Further, data is not collected for commands that expire on entry to Interval Control, because they cannot be canceled (because an element control interval (ICE) is not created). DELAY and POST commands get an EXPIRED response. For START commands there is no such response; so 'expired on entry' is deduced if INTERVAL(0) was specified. This detects most 'expired on entry' STARTs, but not all.

- START, DELAY, and POST commands in error are ignored, so no data is collected.
- CANCEL commands that omit REQID are ignored because they cannot cancel another task. CANCEL commands that return a NOTFND response are also ignored because the ICE must have expired and the CANCEL must have failed. No data is collected for these.
- REQIDs are assumed to be unique; that is, there are no simultaneous pairs of START/CANCEL using the same REQID. Having such a pair violates CICS programming guidelines, and the results from CICS are unpredictable.
- The affinity relation for START might be GLOBAL, BAPPL, LUNAME, or USERID. The lifetime for START might be PCONV, LOGON, SIGNON, ACTIVITY, PROCESS, SYSTEM, or PERMANENT. If the PROTECT option is specified on the START, the lifetime is SYSTEM or PERMANENT. However, the Detector always worsens LOGON and SIGNON to SYSTEM or PERMANENT, because of limitations in the way that this affinity is detected.
- The affinity relation for DELAY and POST might be GLOBAL, BAPPL, LUNAME, or USERID. The Lifetime might be only SYSTEM, PROCESS, ACTIVITY, or PCONV. If the affinity relation is LUNAME or USERID, the lifetime must be PCONV because neither DELAY nor POST exist beyond task termination.
- If the transaction specified on a START or CANCEL command is defined as remote, or a remote SYSID was specified on the command so that the command is function shipped to a remote CICS region, no data is collected. (It is not possible to function ship POST or DELAY commands.)
- A CANCEL affinity is considered to be initiated from a terminal if the START, DELAY or POST is initiated from a terminal. Whether the CANCEL was so initiated or not is irrelevant.

Scanner differences: Scanner detects all instances of POST, all instances of DELAY REQID, all instances of CANCEL REQID, and all instances of START that either omit NOCHECK or specify REQID or specify TERMID (because of RETRIEVE WAIT affinity).

SPI commands

- The commands included here are INQUIRE, SET, CREATE, DISCARD, ENABLE, DISABLE, EXTRACT EXIT, COLLECT STATISTICS, PERFORM, and RESYNC.
- CBTS BROWSE COMMANDS are treated as inquire COMMANDS.
- The affinity here is not an affinity between transactions, but rather an affinity with the system on which the command was issued; that is, a transaction-system affinity. Such affinities do not generate transaction affinity groups, because it does not generally make sense to dynamically route such transactions.
- The use of these commands does require reporting, however, because the system programmer should be aware of the transactions and programs that issue such commands.

Scanner differences: None.

WAIT commands

- The affinity here is really an inter-transaction affinity between the issuer of the WAIT EVENT, WAIT EXTERNAL, or WAITCICS command, and one or more posters. However, the poster of the ECB(s) associated with the WAIT command cannot be detected, because this is not performed via the CICS API. Only half the affinity can be detected.

- This means affinity transaction groups cannot be created, because the affinity degenerates to an affinity with the system on which the WAIT command was issued; that is, a transaction-system affinity.
- The use of WAIT commands does require reporting, however, because the system programmer should be aware of the transactions and programs that issue such commands, and should attempt to locate the posters and so create the correct inter-transaction affinity groups.

Scanner differences: None.

where :

DFHEIV0

Is the slot in working storage into which the argument zero is copied before the BALR to the CICS stub.

D2 Is the decimal offset of the argument zero within the CGT, which you have just calculated.

R2 Is the CGT base register.

When you know the offset of the argument zero within the CGT, you can find the MVC and hence the EXEC CICS command.

An example of finding an EXEC CICS command is given in Figure 56.

For the Load Module Scanner output:

```
CICS INTERDEPENDENCIES ANALYZER
LOAD MODULE SCANNER - DETAILED LISTING OF CICS.DEVR212.LOCLLOAD

Module Name - ACCT04 / Load Module Length - 000159D0 / Module Entry Point
Offset Storage Content (HEX) EDF DEBUG Possibl
-----
000007A6 0A02E0000700004100 00669 WRITEQ
Total possible Dependency cmds = 1
```

The COBOL source after translation was:

```
001123
001124 *EXEC CICS WRITEQ TS QUEUE('ACERLOG') FROM(ACCTERRO)
001125 * LENGTH(ERR-LNG) END-EXEC.
001126 MOVE ' \ 00669 ' TO DFHEIV0
001127 MOVE 'ACERLOG' TO DFHC0080
001128 CALL 'DFHEI1' USING DFHEIV0 DFHC0080 ACCTERRO ERR-LNG
```

The equivalent assembler-language is:

```
001126 MOVE
002764 D210 8558 A6C6 MVC 1368(17,8),1734(10) DFHEIV0
00276A 9240 8569 MVI 1385(8),X'40' DFHEIV0+17
00276E D232 856A 8569 MVC 1386(51,8),1385(8) DFHEIV0+18
001127 MOVE
002774 D207 8340 ACEA MVC 832(8,8),3306(10) DFHC0080
001128 CALL
00277A 4130 8558 LA 3,1368(0,8) DFHEIV0
00277E 5030 D1B0 ST 3,432(0,13) TS2=0
002782 4130 8340 LA 3,832(0,8) DFHC0080
002786 5030 D1B4 ST 3,436(0,13) TS2=4
00278A 4130 75A8 LA 3,1448(0,7) ACCTERRO
00278E 5030 D1B8 ST 3,440(0,13) TS2=8
002792 4130 9A0E LA 3,2574(0,9) ERR-LNG
002796 5030 D1BC ST 3,444(0,13) TS2=12
00279A 9680 D1BC OI 444(13),X'80' TS2=12
00279E 4110 D1B0 LA 1,432(0,13) TS2=0
0027A2 4100 D150 LA 0,336(0,13) CLLE@=2
0027A6 0530 BALR 3,0
0027A8 5030 D158 ST 3,344(0,13) TGT FDMP/TEST-
0027AC 58F0 A000 L 15,0(0,10) V(DFHEI1 )
0027B0 05EF BALR 14,15
0027B2 50F0 D078 ST 15,120(0,13) TGTFIXD+120
0027B6 BF38 D089 ICM 3,8,137(13) TGTFIXD+137
0027BA 0430 SPM 3,0
```

Figure 56. Example of finding an EXEC CICS command from the argument zero

For this example, the calculations are:

```
Load Module Scanner offset    = X'7A6'  
CICS stub length             = X'28'  
Offset of CGT                 = X'B8'  
CGT base register            = GPR 10  
Offset within CGT            = X'7A6' - X'28' - X'B8' = X'6
```

MVC instruction looks like:

```
MVC  d(1,r),1734(10)          DFHEIV0          PGMLIT AT ...'
```

To determine the EXEC CICS command:

1. Look at the assembler language for:

```
MVC  d(1,r),1734(10)          DFHEIV0          PGMLIT AT ...
```

which occurs for the first MOVE:

```
001126  MOVE
```

2. Look at the COBOL source for the MOVE at line 001126. This code is for the EXEC CICS WRITEQ TS command starting on line 001124.

Appendix C. The structure of the CICS IA database

This section describes the following objects:

- “The structure of the Dependency database objects”
- “The structure of the Affinity objects” on page 211
- “The structure of the Load Module Scanner objects” on page 217
- “The structure of the CSECT Scanner objects” on page 219
- “The structure of the CICS regions objects” on page 221
- “The structure of the Command Flow table objects” on page 235
- “The structure of the resource objects” on page 222

The structure of the Dependency database objects

This section describes the tables and views defined in the Dependency database.

- “Dependency base tables”
- “Dependency facilitating tables” on page 208
- “Dependency views” on page 210

Dependency base tables

The Dependency base tables are defined in the database objects. You can write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_CICS_DATA

This table stores information about every unique combination of CICS region, transaction, program, function, and CICS resource recorded by the Collector. With this table you can answer questions such as:

“Which CICS resources are used by this transaction?”

“If I change this CICS resource, which programs and transactions are affected?”

Table 28. The CIU_CICS_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program.
FUNCTION	CHAR(24)	EXEC CICS command name, such as READ, WRITEQ. ^{1 4}
TYPE	CHAR(16)	Resource type, such as TS, program. ^{1 4}
OBJECT	CHAR(255)	Resource name. ²
OBJLENGTH	INTEGER	Length of resource name in OBJECT field.
RMTSYSID	CHAR(4)	Remote SYSID, if relevant. ³
RMTNAME	CHAR(8)	Name by which the resource is known in the remote region.

Table 28. The CIU_CICS_DATA table (continued)

Column	Type	Description
TERMTRAN	CHAR(1)	Whether a terminal is 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.
AFFINITY	CHAR(1)	Whether the EXEC CICS command might cause an affinity: Y Yes N No
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.
COMMAREA	CHAR(1)	Whether a commarea is associated with the transaction: Y Yes N No
CHANNEL	CHAR(1)	Whether a channel is associated with the transaction: Y Yes N No
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Note:

1. If the EIBCODE in the dependency data is not recognized by the database update program, both the FUNCTION and TYPE columns contain ????????
2. The resource name is the name of the object of the function. This column is 50 bytes to accommodate the names of objects such as programs and files, but the data, for example, the 4-byte name of a TD queue, might occupy less space.
3. If the command is shipped or routed to a remote region, these characters tell you the system identifier (SYSID) of the remote region.
4. For information on Function and Type values, see Table 67 on page 239 to Table 103 on page 250.

CIU_DB2_DATA

This table stores information about every unique combination of CICS region, transaction, program, function, and DB2 resource recorded by the Collector. With this table you can answer questions such as:

“Which DB2 resources are used by this transaction?”

“If I change this DB2 resource, which programs and transactions are affected?”

Table 29. The CIU_DB2_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
DB2ID	CHAR(4)	DB2 subsystem ID.
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	CICS program name.
PLAN	CHAR(8)	DB2 plan ID.
RESTYPE	CHAR(16)	DB2 resource type. ¹
RESNAME	CHAR(40)	DB2 resource name.
FUNCTION	CHAR(24)	EXEC SQL command name, such as CREATE, UPDATE. ^{1 2}
SECTION	SMALL INT	The section number, in the source code of the CICS program, at which the DB2 command is issued.
STATEMENT	SMALL INT	The precompiler statement number, in the source code of the CICS program, at which the DB2 command is issued.
TERMTRAN	CHAR(1)	Whether a terminal is associated with the transaction: Y Terminal transaction. N Nonterminal 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, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Note:

1. For more information on RESTYPE and FUNCTION, see Table 107 on page 251.
2. If the command in the dependency data is not recognized by the database update program, the FUNCTION column contains ????????
3. The RESOURCE column might not contain the actual name of the DB2 resource being used. 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.

CIU_IMS_DATA

This table stores information about every unique combination of CICS region,

transaction, program, function, and IMS resource recorded by the Collector. With this table you can answer questions such as:

“Which IMS resources are used by this transaction?”

“If I change this IMS resource, which programs and transactions are affected?”

Table 30. The CIU_IMS_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	Currently active CICS program.
CALLTYPE	CHAR(4)	EXEC for EXEC DLI calls. CBLT for CBLTDLI calls. ASMT for ASMTDLI calls. PLIT for PLITDLI calls.
FUNCTION	CHAR(24)	DLI command. ¹
TYPE	CHAR(16)	PSB or PCB. ¹
OBJECT	CHAR(8)	PSB name or PCB name.
TERMTRAN	CHAR(1)	Whether a terminal is associated with the transaction: Y Terminal transaction. N Nonterminal 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, in the local time format.
LAST_RUN	TIMESTAMP	Time of last occurrence, in the local time format.

Note:

1. For more information on Function and Type, see Table 108 on page 254.

CIU_MQ_DATA

This table stores information about every unique combination of CICS region, transaction, program, function, and MQ resource recorded by the Collector.

With this table you can answer questions such as:

“Which MQ resources are used by this transaction?”

“If I change this MQ resource, which programs and transactions are affected?”

Table 31. The CIU_MQ_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS transaction ID.

Table 31. The CIU_MQ_DATA table (continued)

Column	Type	Description
PROGRAM	CHAR(8)	CICS program name.
FUNCTION	CHAR(24)	MQ command name, such as MQGET, MQPUT ^{1 2} .
TYPE	CHAR(16)	Resource type. ^{1 2}
OBJECT	CHAR(48)	Resource name. ³
QMGRNAME	CHAR(48)	Name of the queue manager.
TERMTRAN	CHAR(1)	Whether a terminal is associated with the transaction: Y Terminal transaction. N Nonterminal 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: Y Yes N No
MQBOO	CHAR(1)	Whether the MQOO_BIND_ON_OPEN option is set for the FUNCTION: Y Yes N No
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format .
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Note:

1. If the command in the dependency data is not recognized by the database update program, both the FUNCTION and TYPE columns contain ????????
2. For more information on Function and Type, see Table 109 on page 254.
3. The resource name is the name of the object of the function. This column has a size of 48 bytes to accommodate the names of objects such as programs and files, but the data might occupy less space.

CIU_NATURAL_DATA

This table stores information about every unique combination of CICS region, transaction, program, function, and Natural resource recorded by the Collector.

Table 32. The CIU_NATURAL_DATA table

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

Table 32. The CIU_NATURAL_DATA table (continued)

Column	Type	Description
FUNCTION	CHAR(8)	Natural command name, such as CALL.
TYPE	CHAR(8)	Natural command type, such as ADABAS, PROGRAM.
OBJECT	CHAR(32)	Database type code (for CALL ADABAS) or calling program name (for CALL PROGRAM).
COMMAND_CODE	CHAR(4)	Database command code (for CALL ADABAS), such as OP, L1, CL.
COMMAND_ID	CHAR(8)	Database command identifier (for CALL ADABAS).
COMMAND_DESC	CHAR(36)	Database command descriptor (for CALL ADABAS).
PROGRAM_TYPE	CHAR(16)	Calling program type (for CALL PROGRAM), such as FETCH PROGRAM, MAP, CALLNAT SUBPROG, VIEW.
PROGRAM_MODE	CHAR(16)	Calling program mode (for CALL PROGRAM), such as STATIC, DYNAMIC.
PROGRAM_CALL	CHAR(16)	Calling program parameter call type (for CALL PROGRAM), such as BY VALUE, BY REFERENCE.
LOCATION	CHAR(8)	Calling program location (for CALL PROGRAM), such as a library name, LIBRARY (load program library), NUCLEUS (Natural shared nucleus).
DATABASE_ID	INTEGER	Database identifier (for CALL ADABAS) or system file database identifier (for CALL PROGRAM).
FILE_NUMBER	INTEGER	File number (for CALL ADABAS) or system file number (for CALL PROGRAM).
TERMTRAN	CHAR(1)	Whether a terminal is associated with the transaction: Y Terminal transaction. N Nonterminal transaction.
TCBMODE	CHAR(2)	CICS TCB in which the application is running; for example, QR or L8.
LEVEL	INTEGER	Current program level.
LINE	CHAR(4)	Current statement number.
OFFSET	CHAR(8)	The offset of the command from the start of the program.
PROGLEN	CHAR(8)	Length of the program (Natural nucleus).
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format

Note: For more information on Function and Type, see Table 110 on page 254.

Dependency facilitating tables

These tables are used by CICS IA to help its processing.

CIU_APPLS_DESC

This table holds the list of applications and a textual description.

Table 33. The CIU_APPLS_DESC table

Column	Type	Description
APPLIC_CODE	CHAR(8)	Application code.
APPLIC_NAME	CHAR(50)	Application Description.

CIU_APPLS_RESOURCES

This table contains all the transactions and programs that make up an application.

Table 34. The CIU_APPLS_RESOURCES table

Column	Type	Description
APPLIC_CODE	CHAR(8)	Application code.
APPLIC_TYPE	CHAR(8)	Resource type (program or transid).
APPLIC_RESNAME	CHAR(32)	Resource name.

Threadsafe table

This table contains a list of CICS commands that are threadsafe in at least one of the supported CICS TS releases. Use the information in this table when reporting which CICS commands used by a program are threadsafe. Load by running job CIUTSLOD in the SCIUSMP2 data set.

CIU_THREADSAFE_CMD

This table stores resource usage information for all resource types for both Web service and program names.

Table 35. The CIU_THREADSAFE_CMD table

Field Name	Type	Description
COMMAND	CHAR(24)	The EXEC CICS command name, such as READ, WRITEQ. Note: The values in this field match those in the COMMAND field in the CIU_SCAN_DETAIL table or the FUNCTION field in the CIU_CICS_DATA table.
RESOURCE_TYPE	CHAR(16)	The resource type, such as TS or PROGRAM. Note: The values in this field match those in the RESOURCE_TYPE field in the CIU_SCAN_DETAIL table or the TYPE field in the CIU_CICS_DATA table.
CICS_TS22	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V2.2. Values are: Y command is threadsafe N command in non-threadsafe I command is indeterminate-threadsaf blank not analyzed to allow for table migration
CICS_TS23	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V2.3. Values are: Y command is threadsafe N command in non-threadsafe I command is indeterminate-threadsaf blank not analyzed to allow for table migration

Table 35. The CIU_THREADSAFE_CMD table (continued)

Field Name	Type	Description
CICS_TS31	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V3.1. Values are: Y command is threadsafe N command in non-threadsafe I command is indeterminate-threadsafe blank not analyzed to allow for table migration
CICS_TS32	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V3.2. Values are: Y command is threadsafe N command in non-threadsafe I command is indeterminate-threadsafe blank not analyzed to allow for table migration
CICS_TS41	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V4.1 Values are: Y command is threadsafe N command in non-threadsafe I command is indeterminate-threadsafe blank not analyzed to allow for table migration

Dependency views

The dependency views show gathered data about dependency.

V_CIU_CICS_INDS

This view is a simple DB2 equi-join of CIU_CICS_DATA, using TRANSID in CIU_CICS_DATA. A selection on FRONT_TXN accesses all the resources, direct and indirect, that this transaction can use. A selection on a resource reports all the FRONT-TXNs that might be affected if the resource is unavailable.

Table 36. View V_CIU_CICS_INDS

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 applid.
HOMESYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS Transaction ID.
PROGRAM	CHAR(8)	Currently active CICS program.
FUNCTION	CHAR(24)	Command, such as READ, LOAD.
TYPE	CHAR(16)	Resource type, such as FILE, TS.
OBJECT	CHAR(50)	Object of function. ¹
RMTSYSID	CHAR(4)	Remote SYSID, if relevant . ²
USECOUNT	INTEGER	Number of times this event has been observed.
FIRST_RUN	TIMESTAMP	Time of first observation of this event, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation of this event, in the local time format.

Note:

1. The column is 50 bytes to accommodate program and file names, but the data might occupy less space.
2. If the command is shipped or routed to a remote region, these characters tell you the system identifier (SYSID) of the remote region.

V_CIU_DB2_RES

This view is a simple DB2 equi-join of CIU_DB2_DATA and SYSIBM.SYSPACKSTMT, using the PROGRAM, SECTION, and STATEMENT fields in CIU_DB2_DATA, and the NAME, SECTNO, and STMTNO fields in SYSIBM.SYSPACKSTMT. You can use it to display the DB2 commands, and therefore the DB2 resources, used by a CICS program.

Table 37. View V_CIU_DB2_RES

Column	Type	Description
APPLID	CHAR(8)	CICS applid.
HOMESYSID	CHAR(4)	SYSID of local region.
DB2ID	CHAR(4)	DB2 subsystem ID.
TRANSID	CHAR(4)	CICS Transaction ID.
PROGRAM	CHAR(8)	CICS program name.
PLAN	CHAR(8)	DB2 plan ID.
SECTNO	SMALLINT	The section number, in the source code of the CICS program, at which the DB2 command is issued .
STMTNO	SMALLINT	The precompiler statement number, in the source code of the CICS program, at which the DB2 command is issued.
STMT	CHAR(*)	The DB2 command (statement) in SYSIBM.SYSPACKSTMT.

V_CIU_DB2_RES2

This view is the same as V_CIU_DB2_RES, except that it joins CIU_DB2_DATA with SYSIBM.SYSSTMT, rather than SYSIBM.SYSPACKSTMT.

The structure of the Affinity objects

This section describes the tables and views defined in the set of Affinity database objects.

- “Affinity base tables”
- “Affinity facilitating table” on page 214
- “Affinity views” on page 214

Affinity base tables

This section describes the Affinity base tables defined in the database. You can write your own SQL applications to query the tables; these applications must use native SQL queries to do this.

CIU_AFF_GRP_DATA

This table stores information about every affinity transaction group; that is, every group of CICS transactions that have been grouped together because they have the potential to create an affinity.

Table 38. The CIU_AFF_GRP_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID.

Table 38. The CIU_AFF_GRP_DATA table (continued)

Column	Type	Description
TRANGROUP	CHAR(10)	Name of the transaction group; for example, TS00000002.
AFFTYPE	CHAR(2)	The type of affinity: IT Intertransaction TS Transaction-system.
GROUPTYPE	CHAR(30)	The group of CICS commands used by this transaction group; 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; one of the following: <ul style="list-style-type: none"> • GLOBAL • BACKGROUND • BAPPL • LINK3270 • LUNAME • USERID.
AFFWORSENEDED	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.
LIFETIME	CHAR(10)	The lifetime of the affinity; one of the following: <ul style="list-style-type: none"> • ACTIVITY = BTS activity • FACILITY = Link3270 bridge facility • LOGON = Logon • PCONV = Pseudoconversation • PERMANENT = Permanent • PROCESS = BTS process • SIGNON = Signon • SYSTEM = System <p>For an explanation of these lifetime values, see “Affinity lifetimes” on page 9.</p>
LIFEWORSENEDED	CHAR(10)	The affinity lifetime has worsened from one of the following: <ul style="list-style-type: none"> • ACTIVITY • FACILITY • LOGON • PCONV • PROCESS • SIGNON • SYSTEM.

Table 38. The CIU_AFF_GRP_DATA table (continued)

Column	Type	Description
RECOVERY	CHAR(1)	Whether the CICS resource is recoverable: Y Recoverable N Not recoverable.
RESOURCE	CHAR(50)	The name of the CICS resource; for example, a program name.
RESLENGTH	INTEGER	Length of resource.
TYPE	CHAR(8)	The type of the CICS resource, such as TS queue, program.
TRANCOUNT	SMALLINT	The total number of CICS transactions in this affinity group.
PROGCOUNT	SMALLINT	The total number of CICS programs in this affinity group.
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. Y This affinity transaction-group is to be included in a "combined" affinity transaction group definition. N This affinity transaction group is not to be included in a "combined" affinity transaction group definition.

CIU_AFF_CMD_DATA

This table records every unique combination of:

- EXEC CICS command with the potential to create an affinity
- Program
- Transaction ID

Table 39. The CIU_AFF_CMD_DATA table

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID
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(24)	EXEC CICS command
RESTYPE	CHAR(16)	Resource type; for example, program
AFFGROUP	CHAR(10)	Name of the affinity transaction group to which this transaction belongs
TERMINAL	CHAR(1)	Whether there is a terminal associated with the transaction: Y Terminal transaction N Nonterminal transaction
BTS	CHAR(1)	Whether this is a BTS task: Y BTS task N Non-BTS task
LINK3270	CHAR(1)	Whether this is a LINK3270 transaction: Y LINK3270 transaction N Non-LINK3270 transaction
USAGE	SMALLINT	Number of times this CICS command is called from this program

Affinity facilitating table

This table is used by CICS IA to help its processing.

CIU_AFF_INDEX_DATA

This table holds the next free index number for the CICS command type, defined by the AFFPREFIX value. It is used, as a numerical suffix, when creating the transaction group (TRANGROUP) name.

Table 40. The CIU_AFF_INDEX_DATA table

Column	Type	Description
AFFPREFIX	CHAR(2)	Prefix indicating a type of CICS command; one of the following: CA CANCEL CO COLLECT CW CWA CR CREATE DI DISCARD EN ENABLE EQ ENQ EX EXTRACT GM GETMAIN GU GETMAIN unmatched IN INQUIRE LD LOAD LF LOAD – FREE LU LOAD – UNLOAD PE PERFORM RE RESYNC RW RETRIEVE TS Temporary storage WA WAIT
AFFINDEX	INTEGER	Next free index number

Affinity views

The following affinity views are defined.

V_CIU_AFFINITY

This view is a simple DB2 equi-join of CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA, using TRANGROUP and APPLID in CIU_AFF_GRP_DATA and AFFGROUP and APPLID in CIU_AFF_CMD_DATA.

Table 41. View V_CIU_AFFINITY

Column	Type	Description
APPLID	CHAR(8)	CICS region APPLID.
TRANGROUP	CHAR(10)	Name of the transaction group, for example, TS00000002.
AFFTYPE	CHAR(2)	The type of affinity: IT Inter transaction TS Transaction system.

Table 41. View V_CIU_AFFINITY (continued)

Column	Type	Description
GROUPTYPE	CHAR(30)	The group of CICS commands used by this transaction, 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, one of the following: <ul style="list-style-type: none"> • GLOBAL • BACKGROUND • BAPPL • LINK3270 • LUNAME • USERID.
AFFWORSENE	CHAR(10)	The relation type of the affinity has worsened from one of the following: <ul style="list-style-type: none"> • BACKGROUND • BAPPL • LINK3270 • LUNAME • USERID <p>For an explanation of these lifetime values, see "Worsening of transaction affinities relations" on page 9.</p>
LIFETIME	CHAR(10)	The lifetime of the affinity, one of the following: <ul style="list-style-type: none"> • ACTIVITY = BTS activity • FACILITY = Link3270 bridge facility • LOGON = Logon • PCONV = Pseudoconversation • PERMANENT = Permanent • PROCESS = BTS process • SIGNON = Signon • SYSTEM = System <p>For an explanation of these lifetime values, see "Affinity lifetimes" on page 9.</p>

Table 41. View V_CIU_AFFINITY (continued)

Column	Type	Description
LIFEWORSENE	CHAR(10)	The affinity lifetime has worsened from, one of the following: <ul style="list-style-type: none"> • ACTIVITY • FACILITY • LOGON • PCONV • PROCESS • SIGNON • SYSTEM For an explanation of these lifeworsened values, see "Worsening of transaction affinities lifetimes" on page 10.
RECOVERY	CHAR(1)	Whether the CICS resource is recoverable: Y Recoverable N Not recoverable.
RESOURCE	CHAR(50)	The name of the CICS resource, for example, a program name.
RESLENGTH	SMALLINT	The length of the name of the CICS resource.
TYPE	CHAR(8)	The type of the CICS resource, such as TS queue, program.
TRANCOUNT	SMALLINT	The total number of CICS transactions in this affinity group.
PROGCOUNT	SMALLINT	The total number of CICS programs in this affinity group.
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. Y This affinity transaction-group is to be included in a "combined" affinity-transaction-group definition. N This affinity transaction-group is not to be included in a "combined" affinity-transaction-group definition.
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	Currently active CICS program.
OFFSET	INTEGER	Offset, from the start of the program, at which this command occurs.
COMMAND	CHAR(24)	EXEC CICS command.
RESTYPE	CHAR(16)	Resource type, for example, program.
TERMINAL	CHAR(1)	Whether there is a terminal associated with the transaction: Y Terminal transaction N Nonterminal transaction.
BTS	CHAR(1)	Whether this is a BTS task: Y BTS task N Non-BTS task.
LINK3270	CHAR(1)	Whether this is a LINK3270 transaction: Y LINK3270 transaction N Non-LINK3270 transaction.

The structure of the Load Module Scanner objects

This section describes the tables and views defined in the Load Module Scanner database objects.

- “Load Module Scanner base tables”

Load Module Scanner base tables

This section describes the Load Module Scanner base tables defined in the database. You can write your own SQL applications to query the tables; these applications must use native SQL queries to do this.

CIU_SCAN_SUMMARY

This table stores summary information about every module in the load libraries that have been scanned.

Table 42. The CIU_SCAN_SUMMARY table

Column	Type	Description
DSNAME	CHAR(44)	Data set name
PROGRAM	CHAR(8)	Module name
LANGUAGE	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

CIU_SCAN_DETAIL

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

Table 43. The CIU_SCAN_DETAIL table

Column	Type	Description
DSNAME	CHAR(44)	Data set name
PROGRAM	CHAR(8)	Module name
OFFSET	INTEGER	Offset, from the start of the program, at which this command occurs
COMMAND	CHAR(24)	EXEC CICS command or MVS POST
RESOURCE_TYPE	CHAR(16)	Resource type; for example, program
AFFINITY	CHAR(1)	Whether this command has the potential to create an affinity: Y Yes N No
AFFINITY_TYPE	CHAR(2)	The type of affinity: IT Inter-transaction TS Transaction-system

Table 43. The CIU_SCAN_DETAIL table (continued)

Column	Type	Description
DEPENDENCY	CHAR(1)	Whether this command has the potential to create a dependency: Y Yes N No
MVS_POST	CHAR(1)	Whether this command is a possible MVS POST: Y Yes N No
COMMAND_HEX	CHAR(50)	Data at the command offset, shown in hexadecimal

V_CIU_SCAN_TRDSAFE

This view is a simple join between the CIU_SCAN_DETAIL table and the CIU_THREADSafe_CMD table using the COMMAND and RESOURCE_TYPE fields from each table. This table is used to query, by CICS TS release, which commands in the CIU_SCAN_DETAIL table are threadsafe, non-threadsafe, or indeterminate-threadsafe.

Table 44. The V_CIU_SCAN_TRDSAFE table

Column	Type	Description
DSNAME	CHAR(44)	Data set name
PROGRAM	CHAR(8)	Module name
LANGUAGE	CHAR(10)	Programming language detected
LE	CHAR(7)	Language Environment (LE) detected
CICS_OR_BATCH	CHAR(5)	CICS transaction or batch
AFFINITY_COUNT	INTEGER	Resource type; for example, program
MVS_POST_COUNT	INTEGER	Number of MVS POST commands
DEPENDENCY_COUNT	INTEGER	Number of commands with potential to create dependencies
CICS_TS22	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V2.2. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS23	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V2.3. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS31	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V3.1. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS32	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V3.2. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe

The structure of the CSECT Scanner objects

This section describes the tables and views defined in the CSECT Scanner database.

It contains:

- “CSECT Scanner base tables”
- “CSECT Scanner object views” on page 220

CSECT Scanner base tables

Use the information in the CSECT Scanner base tables defined in the database to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_PROGRAM_INFO

This table stores load-module-related information from the load libraries that have been scanned. It allows you to answer questions such as:

“Given a load module name and length, when was it link-edited?”

“Is a program resource a GLUE or TRUE program?”

Table 45. The CIU_PROGRAM_INFO table

Column	Type	Description
DSNAME	CHAR(44)	Data set name.
PROGRAM	CHAR(8)	Load module name.
PROGLEN	CHAR(8)	Load module length in hexadecimal.
ENTRY_POINT	CHAR(8)	Entry point offset in hexadecimal.
ALIAS_OF	CHAR(8)	If the program name is an alias, this program is the one for which it is an alias.
LINKER_NAME	CHAR(10)	Identifier of the binder or link-editor.
LINKER_VERSION	CHAR(5)	Version number of the binder or link-editor (VV.RR).
LINKED	TIMESTAMP	Date and local time that the program was bound or link-edited.
AMODE	CHAR(3)	Addressing mode.
RMODE	CHAR(3)	Residence mode.

CIU_CSECT_INFO

This table stores CSECT-related information from the load libraries that have been scanned. It allows you to answer questions such as:

“Given a load module name and length, what is the user data of the CSECT with the same name as the load module?”

Table 46. The CIU_CSECT_INFO table

Column	Type	Description
DSNAME	CHAR(44)	Data set name.
PROGRAM	CHAR(8)	Load module name.
PROGLEN	CHAR(8)	Load module length in hexadecimal.
LINKED	TIMESTAMP	Date and local time that the program was bound or link-edited.
CSECT_NAME	CHAR(8)	CSECT name.

Table 46. The CIU_CSECT_INFO table (continued)

Column	Type	Description
TRAN_1_DATE	CHAR(7)	First translation date (YYYYDDD).
TRAN_1_NAME	CHAR(10)	First translator identifier.
TRAN_1_VERSION	CHAR(5)	First translator version (VV.RR).
TRAN_2_DATE	CHAR(7)	Second translation date (YYYYDDD).
TRAN_2_NAME	CHAR(10)	Second translator identifier.
TRAN_2_VERSION	CHAR(5)	Second translator version (VV.RR).
USER_DATA_DATE	CHAR(7)	User data date (YYYYDDD).
USER_DATA	VARCHAR(80)	User data.
HMASPZAP_DATE	CHAR(7)	ZAP date (YYYYDDD).
HMASPZAP_DATA	CHAR(8)	ZAP data.

Note:

1. Some CSECTs are translated, or compiled, twice. For example, programs created using the IBM internal PL/X language are first translated by the PL/X compiler and then translated by the assembler-program.
2. Dates are in the format “YYYYDDD”, where YYYY is the year and DDD is the day of the year. For example: “2007124”.

CIU_TRANSLATORS

This table stores descriptions of translators, binders, and linkage editors. It is loaded with predefined information about a range of IBM products by the CIUTLOAD job. It allows you to determine the full descriptions of these programs from the identifiers used in the tables above.

Table 47. The CIU_TRANSLATORS table

Column	Type	Description
TRANSLATOR_NAME	CHAR(10)	The translator identifier.
DESCRIPTION	CHAR(64)	Description of translator.

CSECT Scanner object views

The CSECT Scanner object views show scanned data about programs and dependency.

V_CIU_CICS_LINKED

This view is a simple DB2 equi-join of CIU_CICS_DATA and CIU_PROGRAM_INFO, using PROGRAM and PROGLN. It shows the link-edit timestamps of the programs that are using CICS resources.

Table 48. View V_CIU_CICS_LINKED

Column	Type	Description
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	Load module name.
LINKED	TIMESTAMP	Bind or link-edit timestamp, in the local time format.
FUNCTION	CHAR(8)	Name of EXEC CICS command.

Table 48. View V_CIU_CICS_LINKED (continued)

Column	Type	Description
TYPE	CHAR(8)	Resource type.
OBJECT	CHAR(255)	Resource name.

V_CIU_CSECT_TRANS

This view is a simple DB2 equi-join of CIU_CSECT_INFO and CIU_TRANSLATORS using TRAN_1_NAME in CIU_CSECT_INFO and TRANSLATOR_NAME in CIU_TRANSLATORS. It shows the descriptions of the compilers used to create the CSECTs in each scanned load module.

Table 49. View V_CIU_CSECT_TRANS

Column	Type	Description
DSNAME	CHAR(44)	Data set name.
PROGRAM	CHAR(8)	Load module name.
LINKED	TIMESTAMP	Bind or link-edit timestamp, in the local time format.
CSECT_NAME	CHAR(8)	CSECT name.
TRAN_1_NAME	CHAR(10)	Compiler ID.
DESCRIPTION	CHAR(64)	Compiler description.

The structure of the CICS regions objects

This section describes the structure of the CICS regions objects.

- “CICS regions base table”

CICS regions base table

Use the information in the CIU_REGION_INFO table in the CICS regions base table defined in the database to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_REGION_INFO

This table stores information about the CICS regions on which the Collector has run. Whenever a collection of dependency or affinity data is started, the names of the CICS System Definition data set (CSD) and of the first four resource group lists in the CSD are stored, together with date and time information.

Table 50. The CIU_REGION_INFO table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	CICS region system identifier (SYSID).
CSD_RELEASE	CHAR(4)	The CICS level number for the CICS region.
CSD_NAME	CHAR(44)	Name of the CICS System Definition data set (CSD) used during the last collection in this region.
CSD_GROUP_LIST1	CHAR(8)	Name of the first resource group list in the CSD defined at system startup.
CSD_GROUP_LIST2	CHAR(8)	Name of the second resource group list in the CSD defined at system startup.

Table 50. The CIU_REGION_INFO table (continued)

Column	Type	Description
CSD_GROUP_LIST3	CHAR(8)	Name of the third resource group list in the CSD defined at system startup.
CSD_GROUP_LIST4	CHAR(8)	Name of the fourth resource group list in the CSD defined at system startup.
STORAGE_PROTECT	CHAR(8)	Indicates if storage protection was active for the region.
DEP_COLL_LASTSTART	TIMESTAMP	Time at which the last collection of dependency data was started, in the local time format.
DEP_COLL_LASTSAVE	TIMESTAMP	Time at which collected dependency data was last saved, in the local time format.
APP_COLL_LASTSTART	TIMESTAMP	Time at which the last collection of affinity data was started, in the local time format.
APP_COLL_LASTSAVE	TIMESTAMP	Time at which collected affinity data was last saved, in the local time format.

The structure of the resource objects

This section describes the structure of the resource objects.

- “File resource table”
- “Program resource table” on page 224
- “Transaction resource table” on page 225
- “Transient Data queue resource table” on page 228
- “Temporary Storage queue resource table” on page 230
- “Web service resource table” on page 231
- “Threadsafe table” on page 209
- “GLUE and TRUE exit resource table” on page 232

File resource table

Use the information in the CIU_FILE_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_FILE_DETAIL

This table stores detailed information about every CICS file referenced in a transaction recorded by the Collector. File information is stored in this table only if the **Files** field on the CICS Resources Options panel, CIU240, is set to D.

Table 51. The CIU_FILE_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local 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. Values are: BLOCKED, UNBLOCKED.
BLOCKKEYLEN	INTEGER	Physical block key length for the file.

Table 51. The CIU_FILE_DETAIL table (continued)

Field Name	Type	Description
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. Values are: OLD, 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. Values are: LOAD, NOLOAD, and NOTAPPLIC.
LSRPOOLID	INTEGER	The number of VSAM LSR pool associated with this file.
MAXNUMRECS	INTEGER	The maximum number of records that the file can hold.
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. Value are as follow: 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 whether the records can be read from the file. Values are: NOTREADABLE or READABLE.
RECORDFORMAT	CHAR(10)	Identifies the format of the records on the file. Values are: FIXED, VARIABLE, or UNDEFINED.
RECORDSIZE	INTEGER	The size of a fixed-length record or the maximum size of a variable-length record.
RECOVSTATUS	CHAR(14)	Indicates whether the file is recoverable. 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. Value are as follows: 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 named in the REMOTESYSTEM field.
REMOTESYSTEM	CHAR(4)	The name of the CICS region in which the file is defined.
REMOTETABLE	CHAR(8)	Indicates whether the file represents an open remote data table. Value is: REMTABLE (an open remote data table).
RLSACCESS	CHAR(10)	Indicates whether the file is defined to be opened in RLS mode. Values are: NOTAPPLIC, NOTRLS, or RLS.
STRINGS	INTEGER	Indicates the number of strings (concurrent operations) specified for the file.
TABLE	CHAR(10)	Identifies the type of data set that corresponds to this file. Values are: CFTABLE, CICSTABLE, NOTTABLE, or USERTABLE.
TABLERNAME	CHAR(8)	The name specified for the coupling facility data table.

Table 51. The CIU_FILE_DETAIL table (continued)

Field Name	Type	Description
TYPE	CHAR(10)	The type of data set that corresponds to this file. Values are: ESDS, KEYED, KSDS, NOTKEYED, RRDS, VRRDS, NOTAPPLIC.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Program resource table

Use the information in the CIU_PROGRAM_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_PROGRAM_DETAIL

This table stores information about every CICS program referenced in a transaction recorded by the Collector. Program information is stored in this table only if the **Programs** field on the CICS Resources Options panel, CIU240, is set to D.

Table 52. The CIU_PROGRAM_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PROGRAM_NAME	CHAR(8)	Name of the program resource.
LINKEDIT_DATE	TIMESTAMP	Reserved for future use.
LIB_NAME	CHAR(8)	Name of the library resource from which this program was loaded.
LIB_DATASET-NAME	CHAR(44)	Name of the data set from which the program was loaded.
ACCESS	CHAR(8)	The type of storage into which the program is loaded. Values are: CICS, USER, READONLY, and NONE.
APIST	CHAR(8)	Indicates the API attribute of the installed program definition. Values are: CICSAPI and OPENAPI.
CONCURRENCY	CHAR(12)	Indicates the concurrency attribute of the installed program definition. Values are: QUASIRENT and THREADSAFE.
DATA_LOCATION	CHAR(10)	Indicates whether this program can accept data address higher than 16 MB. Values are: ANY, BELOW, and NOTAPPLIC.
DYNAMIC_STATUS	CHAR(10)	Indicates if the program is the subject of a program-link request; that is, the request can be dynamically routed. Values are: DYNAMIC and NOTDYNAMIC.
EXECUTION_KEY	CHAR(10)	Indicates the storage key of the program. Values are: CICS, USER, and NOTAPPLIC.
EXECUTION_SET	CHAR(10)	Indicates whether the program is restricted to the distributed program link subset of the CICS API. Values are: DPLSUBSET, FULLAPI, and NOTAPPLIC.
HOLD_STATUS	CHAR(10)	Indicates how long the program is to remain loaded. Values are: CICSLIFE, TASKLIKE, and NOTAPPLIC.
INSTALL_TYPE	CHAR(10)	Indicates the method used to install the program resource definition. Values are: AUTO, CATALOG, GROUPLIST, MANUAL, RDO, and SYSAUTO.

Table 52. The CIU_PROGRAM_DETAIL table (continued)

Field Name	Type	Description
LANGUAGE_DEDEDUCED	CHAR(12)	Indicates the language deduced by CICS for the program. Values are: ASSEMBLE, C370, COBOL, COBOL2, LE370, PLI, JAVA, NOTDEDUCED, and NOTAPPLIC.
LANGUAGE_DEFINED	CHAR(12)	Indicates the programming language specified on the resource definition. Values are: ASSEMBLE, C370, COBOL, LE370, PLI, NOTDEFINED, and NOTAPPLIC.
LOAD_STATUS	CHAR(12)	Indicates whether the program can be loaded. Values are: LOADABLE, NOTLOADABLE, NOTLOADED, and NOTAPPLIC.
LOCATION	CHAR(8)	Indicates where the most recently loaded copy of the program resides. Values are: CDSA, ECDSA, ELPA, ERDSA, ESDSA, LPA, RDSA, SDSA, and NONE
MODULE_TYPE	CHAR(12)	Indicates the type of this program resource. Values are: MAPSET, PARTITIONSET, and PROGRAM.
PROGRAM_ATTRIBUTE	CHAR(10)	Indicates the residency status of the program. Values are: RELOAD, RESIDENT, REUSABLE, TRANSIENT, and TEST.
PROGRAM_LENGTH	INTEGER	The length of the program in bytes.
PROGRAM_TYPE	CHAR(10)	Indicates where the next new copy of the program is to be loaded from. Values are: PRIVATE, SHARED, TYPEANY, and NOTAPPLIC.
PROGRAM_USAGE	CHAR(12)	Indicates whether the program is used as a CICS nucleus program or as a user application program. Values are: APPLICATION or NUCLEUS.
REMOTE_DEFINITION	CHAR(8)	Indicates whether this program is a local or a remote resource. Values are: LOCAL or REMOTE.
REMOTE_PROGID	CHAR(8)	The name by which this program is known in the remote CICS region.
REMOTE_SYSID	CHAR(4)	The name of the remote CICS region that owns the program.
REMOTE_TRANID	CHAR(4)	The name of the transaction under which the program runs on the remote CICS region.
SPECIFIED_AMODE	CHAR(12)	The addressing mode specified for the resource. Values are: AMODE24, AMODE31, AMODEANY, and NOTSPECIFIED.
SPECIFIED_RMODE	CHAR(12)	The residency mode specified for the resource. Values are: RMODE24, RMODE31, RMODEANY, and NOTSPECIFIED.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Transaction resource table

Use the information in the CIU_TRANSID_DETAIL tables to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_TRANSID_DETAIL

This table stores detailed information about every CICS transaction recorded by the Collector. Transaction information is stored in this table only if the **Transactions** field on the CICS Resources Options panel, CIU240, is set to D.

Table 53. The CIU_TRANSID_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region
TRANSID	CHAR(4)	The name of the transaction definition.
BREXIT	CHAR(8)	The name of the bridge exit defined by the BREXIT parameter of the named transaction definition.
CMDSEC	CHAR(1)	Indicates whether command security checking will be performed for the tasks running the transaction. Values are: Y Yes N No
DTIMEOUT	INTEGER	The deadlock time-out value in seconds for the task running this transaction.
DUMP	CHAR(1)	Indicates whether the transaction is defined for dumping. Values are: Y Yes N No
DYNAMIC	CHAR(1)	Indicates whether the transaction is defined for dynamic transaction routing. Values are: Y Yes N No
INDOUBT	CHAR(8)	The action CICS takes if the CICS region fails or loses connectivity with its coordinator while a unit of work is in the indoubt period.
INDOUBT_WAIT	CHAR(1)	The response that the CICS unit of work takes if a failure occurs while in an indoubt state. Values are: Y The UOW is to wait, pending recovery from a failure, to determine whether recoverable resources are to be backed out or committed. N The UOW is not to wait. CICS immediately takes the action specified on the ACTION attribute of the TRANSACTION definition.
INDOUBT_WAIT_TIME	INTEGER	The length of time, in minutes, after a failure during the indoubt period, before the transaction takes the action returned in the INDOUBT field.
INITIAL_PROGRAM	CHAR(8)	The name of the initial program given control for the transaction.
ISOLATE	CHAR(1)	Indicated whether transaction isolation is required for the transaction task-lifetime user-key storage. Values are: Y Transaction isolation is required. N Transaction isolation is not required.
LOCAL_QUEUEING	CHAR(1)	Indicates whether the started request for this transaction is eligible to be queued locally if the transaction is to be started on another system and the remote system is not available. Values are: Y The request can be queued locally. N The request is not queued locally.
OTSTIMEOUT	INTEGER	The period of time, in seconds, that an Object Transaction Service (OTS) transaction is allowed to run without the initiator of the OTS transaction taking a sync point (or rolling back the OTS transaction).

Table 53. The CIU_TRANSID_DETAIL table (continued)

Field Name	Type	Description
PARTITIONSET	CHAR(8)	Indicates the partition set specified on the transaction definition. Values are: KEEP, NAMED, OWN, and NONE.
PARTITIONSET_NAME	CHAR(8)	The partition set defined on the transaction definition.
PROFILE_NAME	CHAR(8)	The profile definition associated with the transaction definition.
REMOTE	CHAR(1)	Indicates whether the transaction is defined as remote. Values are: Y Yes N No
REMOTE_NAME	CHAR(8)	The remote name as specified on the transaction definition.
REMOTE_SYSTEM	CHAR(4)	The remote system as specified on the transaction definition.
RESSEC	CHAR(1)	Indicates whether resource security checking is required for the transaction. Values are: Y Yes N No
RESTART	CHAR(1)	Indicates whether the transaction is considered for transaction restart. Values are: Y The transaction can be restarted. N The transaction can not be restarted.
ROUTABLE_STATUS	CHAR(12)	Indicates whether the transaction, from a START command, is routed using the enhanced routing method. Values are: ROUTABLE and NOTROUTABLE.
RUNAWAY_LIMIT	INTEGER	The runaway-task time limit specified on the transaction definition.
SHUTDOWN	CHAR(8)	Indicates whether the transaction can be run during CICS shutdown. Values are: ENABLED and DISABLED.
SPURGE	CHAR(1)	Indicates whether the transaction is defined as system-purgeable. Values are: Y Yes N No
STORAGE_CLEAR	CHAR(1)	Indicates whether task-lifetime storage is cleared before it is freed by a FREEMAIN command. Values are: Y Yes N No
STORAGE_FREEZE	CHAR(1)	Indicates whether the storage freeze option is defined for the transaction. Values are: Y Yes N No
SYSTEM_ATTACH	CHAR(1)	Indicates whether the tasks attached with this transaction are attached as system tasks. Values are: Y Yes N No
SYSTEM_RUNAWAY	CHAR(8)	Indicates whether the transaction is governed by the system runaway limit. Values are: Y Yes N No

Table 53. The CIU_TRANSID_DETAIL table (continued)

Field Name	Type	Description
TASKDATAKEY	CHAR(8)	The storage key for the task-lifetime storage associated with the transaction. Values are: CICS and USER.
TASKDATALOC	CHAR(8)	The storage location for the task-lifetime storage associated with the transaction. Values are: Any The storage can be located above 16 MB in virtual storage. Below The storage must be located below 16 MB in virtual storage.
TCLASS	CHAR(1)	Indicates whether the transaction belongs to a transaction class. Values are: Y Yes N No
TCLASS_NAME	CHAR(8)	The name of the transaction class that the transaction belongs to.
TPURGE	CHAR(1)	Indicates whether the transaction is purgeable in the event of a VTAM terminal error. Values are: Y Yes N No
TRACE	CHAR(8)	The level of tracing defined for the transaction. Values are: SPECIAL, STANDARD, and SUPPRESSED.
TRAN_ROUTING_PROFILE	CHAR(8)	The name of the profile CICS uses to route the transaction to a remote system.
PRIMARY_TRANSID	CHAR(4)	The primary transaction identifier for the transaction definition.
TWASIZE	INTEGER	The size of the transaction work area specified on the transaction definition.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Transient Data queue resource table

Use the information in the CIU_TDQUEUE_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_TDQUEUE_DETAIL

This table stores detailed information about every CICS transient data queue referenced in a transaction recorded by the Collector. Transient data queue information is stored in this table only if the **TD Queues** field on the CICS Resources Options panel, CIU240, is set to D.

Table 54. The CIU_TDQUEUE_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
TDQUEUE_NAME	CHAR(4)	The name of the transient data queue definition.
ATIFACILITY	CHAR(10)	Indicates whether the queue has a terminal or session associated with it. Values are: NOTAPPLIC, NOTERMINAL, and TERMINAL
ATITERMID	CHAR(4)	The name of the terminal or session associated with the queue.

Table 54. The CIU_TDQUEUE_DETAIL table (continued)

Field Name	Type	Description
ATITRANID	CHAR(4)	Identifies the transaction to be run when CICS initiates a task automatically to process the queue.
ATIUSER	CHAR(8)	The user identifier associated with the queue.
BLOCKFORMAT	CHAR(10)	Indicates whether the data set associated with the queue is in blocked record format or not. Values are: BLOCKED, UNBLOCKED, and NOTAPPLIC.
BLOCKSIZE	INTEGER	The length of the block in bytes.
DATABUFFERS	INTEGER	The number of buffers to be used by this queue.
DDNAME	CHAR(8)	An identifier that refers to an associated data set name.
DISPOSITION	CHAR(10)	Indicates the status of the associated data set. Values are: MOD, OLD, SHARE, and NOTAPPLIC.
DSNAME	CHAR(44)	The name of the associated QSAM data set.
ERROROPTION	CHAR(8)	Indicates the action CICS takes if an I/O error is encountered. Values are: IGNORERR The block that caused the error is accepted. SKIP The block that caused the error is skipped.
INDIRECTNAME	CHAR(4)	The name of the queue to which this indirect queue points.
INDOUBT	CHAR(8)	Indicates the action that CICS takes for an indoubt unit of work, if the definition for this queue specifies WAIT(YES). Values are: QUEUE and REJECT.
INDOUBTWAIT	CHAR(8)	Indicates whether an indoubt unit of work will wait for resynchronization with its coordinator to determine whether to commit or back out the changes. Values are: NOWAIT and WAIT.
IOTYPE	CHAR(10)	Indicates whether the queue is defined for input or output. Values are: INPUT, OUTPUT, RDBACK, or NOTAPPLIC.
MEMBER	CHAR(8)	Member name if the queue is a member of a partitioned data set.
PRINTCONTROL	CHAR(10)	Indicates the type of print control, if any, defined for the queue. Values are: ASACTL, MCHCTL, NOCTL, and NOTAPPLIC.
RECORDFORMAT	CHAR(10)	Indicates whether the queue has fixed-length or variable-length records. Values are: FIXED VARIABLE, and NOTAPPLIC.
RECORDLENGTH	INTEGER	The record length, in bytes, for queues having fixed-length records, or the maximum record length of queues having variable-length records. Applies only to extrapartition queues; for others, -1 is present.
RECOVSTATUS	CHAR(12)	Indicates the type of recovery defined for the queue. Recovery is available only for intrapartition queues. Values are: LOGICAL, PHYSICAL, NOTRECOVABLE, and NOTAPPLIC.
REMOTENAME	CHAR(4)	The name of the queue in the remote CICS region in which the queue is defined. Applies only to queues defined as remote, for other queues the value is blanks.

Table 54. The CIU_TDQUEUE_DETAIL table (continued)

Field Name	Type	Description
REMOTESYSTEM	CHAR(4)	The name of the CICS region in which the queue is defined. Applies only to queues defined as remote; for other queues the value is blanks.
REWIND	CHAR(8)	Indicates the disposition of a tape data set. Values are: LEAVE and REREAD.
SYSOUTCLASS	CHAR(1)	Indicates the class attribute of the associated SYSOUT data set, or blank if DSNAME is used.
TRIGGERLEVEL	INTEGER	The number of items the queue must contain before automatic transaction initiation (ATI) occurs. A value of zero means the queue is not subject to ATI. A value of -1 means the queue is not intrapartition.
TYPE	CHAR(8)	Identifies the type of queue. Values are: EXTRA, INDIRECT, INTRA, and REMOTE.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Temporary Storage queue resource table

Use the information in the CIU_TSQUEUE_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_TSQUEUE_DETAIL

This table stores detailed information about every CICS temporary storage queue referenced in a transaction recorded by the Collector. TS Queue information is stored in this table only if the **TS Queues** field on the CICS Resources Options panel, CIU240, is set to D.

Table 55. The CIU_TSQUEUE_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
TSQUEUE_NAME	CHAR(16)	Name of the temporary storage queue.
FLENGTH	INTEGER	The total length in bytes of all items in the temporary storage queue.
LOCATION	CHAR(10)	Indicates where the temporary storage queue resides. Values are: AUXILIARY The queue is held in CICS temporary storage VSAM data sets. MAIN The queue is held in main storage.
MAXITEMLEN	INTEGER	The length in bytes of the largest item in the queue.
MINITEMLEN	INTEGER	The length in bytes of the smallest item in the queue.
POOLNAME	CHAR(8)	The name of a temporary storage pool. CICS ships the command to the temporary storage server that manages the pool.
RECOVSTATUS	CHAR(14)	Indicates the recovery status of the queue. Values are: RECOVERABLE and NOTRECOVERABLE.
SYSID	CHAR(8)	The system name that corresponds to a temporary storage pool name.

Table 55. The CIU_TSQUEUE_DETAIL table (continued)

Field Name	Type	Description
TRANSID	CHAR(4)	Identifies the transaction that created the temporary storage queue.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

Web service resource table

Use the information in the CIU_WEBSERV_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_WEBSERV_DETAIL

This table stores detailed information about every CICS Web service resource referenced in a transaction recorded by the Collector. Web service information is stored in this table only if the **Web Services** field on the CICS Resource Options panel, CIU240, is set to D.

Table 56. The CIU_WEBSERV_DETAIL table

Field Name	Type	Description
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local region.
NAME	CHAR(32)	The name of the Web service.
PROGRAM	CHAR(8)	The name of the CICS program that implements the Web service.
URIMAP	CHAR(8)	The name of the dynamically installed URIMAP.
CCSID	CHAR(8)	The CCSID that is used to encode the character data in the application data structure at run time.
CONTAINER	CHAR(16)	The name of the container used if PGMINTERFACE contains a value of CHANNEL.
MAPPINGLEVEL	CHAR(8)	The mapping level that is used to convert data between language structures and Web service description (WSDL) documents. Values are 1.0, 1.1, 1.2, 2.0, or 2.1.
MAPPINGRNUM	INTEGER	The release number for the mapping level that is used to convert data between language structures and Web services description (WSDL) documents. Values are: 0, 1, or 2.
MAPPINGVNUM	INTEGER	The version number of the mapping level that is used to convert data between language structures and Web service description (WSDL) documents. Values are: 1 or 2.
MINRUNLEVEL	CHAR(8)	The minimum runtime level that is required to run the Web service in CICS. Values are 1.0, 1.1, 1.2, 2.0, or 2.1.
MINRUNRNUM	INTEGER	The release number for the minimum runtime level that is required to run the Web services in CICS. Values are: 0, 1, or 2.
MINRUNVNUM	INTEGER	The version number for the minimum runtime level that is required to run the Web services in CICS. Values are: 0, 1, or 2.

Table 56. The CIU_WEBSERV_DETAIL table (continued)

Field Name	Type	Description
PIPELINE	CHAR(8)	The name of the PIPELINE resource that contains this WEBSERVICE resource.
PGMINTERFACE	CVDA	Indicates whether the CICS program that implements the Web service expects input in a channel or in a commarea. Values are: CHANNEL or COMMAREA.
VALIDATIONSTATUS	CHAR(12)	Indicates whether full validation of SOAP messages is currently enabled for this WEBSERVICE. Values are: VALIDATION or NOVALIDATION.
XOPDIRECTST	CHAR(12)	Indicates whether the Web service is currently able to handle XOP documents in direct mode. Values are: NOXOPDIRECT or XOPDIRECT.
XOPSUPPORTST	CHAR(12)	Indicates whether the Web service implementations is capable of handling XOP documents and binary attachments in direct mode. Values are: NOXOPSUPPORT or XOPSUPPORT.
WSDL_FILENAME	CHAR(255)	The name of the Web service description file associated with the WEBSERVICE resource.
WSBIND_FILENAME	CHAR(255)	The name of the Web service binding file.
ENDPOINT	CHAR(255)	The endpoint URI of a remote WEBSERVICE.
BINDING	CHAR(255)	The WSDL binding represented by the WEBSERVICE.
LAST_MODIFIED	TIMESTAMP	The time the deployed WSBind file on z/OS UNIX was last updated, in the local time format.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

GLUE and TRUE exit resource table

Use the information in the CIU_EXIT_INFO tables to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_EXIT_INFO

This table stores detailed information about every CICS GLUE and TRUE exit that is called by at least one transaction. Exit information is stored in this table only if the **Exits** field on the CICS Resources Options panel, CIU240, is set to Y.

Table 57. The CIU_EXIT_INFO table

Field Name	Type	Special restrictions
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
EXIT_PROGRAM	CHAR(8)	The name of the exit program.
EXIT_NAME	CHAR(8)	The name of the exit.
EXIT_POINT	CHAR(8)	The name of the entry point associated with the exit. GLUEs only.
EXIT_TYPE	CHAR(4)	The type of exit. Values are GLUE or TRUE.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

CIU_TRUEEXIT_INFO

Table 58. The CIU_TRUEEXIT_INFO table

Field Name	Type	Special restrictions
TRUE_NAME	CHAR(8)	The TRUE exit program name.
PRODUCT_INFO	CHAR(50)	The product name.

V_CIU_TRUEEXIT_INFO

Table 59. The V_CIU_TRUEEXIT_INFO table

Field Name	Type	Special restrictions
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	Currently active CICS program.
FUNCTION	CHAR(24)	CALL.
TYPE	CHAR(24)	EXIT.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.
TRUE_NAME	CHAR(8)	The TRUE exit program name.
PRODUCT_INFO	CHAR(50)	The product name.

Event table

Use the information in the CIU_EVENT_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_EVENT_DETAIL

This table stores detailed EVENT information.

Table 60. The CIU_EVENT_DETAIL table

Column	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
EVENT_NAME	CHAR(32)	Event name.
CAPTURE_SPEC	CHAR(32)	Capture Specification name.
EVENT_BINDING	CHAR(32)	Event Binding name.
EVENT_TYPE	CHAR(1)	Event type (application or system).
EVENT_STRUCID	CHAR(4)	Event structure identifier.
EVENT_VERSION	INTEGER	Event Structure version number.
SCHEMA_VER	INTEGER	Schema version number.
SCHEMA_REL	INTEGER	Schema release number.
EVENT_USERTAG	CHAR(8)	Event Binding user tag.
EB_DEF_SOURCE	CHAR(8)	Event Binding definition source. The source of the definition, depending on which, agent made the last change.
EB_DEF_TIME	TIMESTAMP	Event Binding definition time. The local date and time when the resource definition record was created.

Table 60. The CIU_EVENT_DETAIL table (continued)

Column	Type	Description
EB_STATUS	CHAR(12)	Event Binding status. Indicates whether the event binding is enabled or not.
EB_ADAPTER	CHAR(32)	Event Binding Adapter.
EB_CHG_AGENT	CHAR(12)	Event Binding change agent identifier.
EB_CHG_REL	CHAR(4)	The CICS release level of the agent that made the last modification.
EB_CHG_TIME	TIMESTAMP	The local date and time when the definition was last changed.
EB_CHG_USERID	CHAR(8)	The user ID that made the last modification.
EB_INST_AGENT	CHAR(12)	The install agent identifier that made the installation.
EB_INST_TIME	TIMESTAMP	The local date and time when the definition was installed.
EB_INST_USERID	CHAR(8)	The user ID that installed the resource definition.
BUNDLE_BASESCOPE	CHAR(255)	Base scope of bundle.
BUNDLE_DIR	CHAR(255)	Name of the BUNDLE directory.
BUNDLE_CHG_AGENT	CHAR(12)	Last modification agent.
BUNDLE_CHG_AGREL	CHAR(4)	Last modification agent release.
BUNDLE_CHG_TIME	TIMESTAMP	Last modification time.
BUNDLE_CHG_USERID	CHAR(8)	Last modification user ID.
BUNDLE_DEF_SOURCE	CHAR(8)	Source of the resource definition.
BUNDLE_DEF_TIME	TIMESTAMP	Creation time.
BUNDLE_ENA_COUNT	INTEGER	Enabled count.
BUNDLE_ENA_STATUS	CHAR(12)	Status.
BUNDLE_INST_AGENT	CHAR(12)	Installation agent.
BUNDLE_INST_TIME	TIMESTAMP	Installation time.
BUNDLE_INST_USERID	CHAR(8)	Installation user ID.
BUNDLE_PART_COUNT	INTEGER	Part count.
BUNDLE_TARGET_COUNT	INTEGER	Target count.
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.
IN_db2dbnt_CIUEVNT		
AUDIT		
CCSID EBCDIC		

The structure of the CICS IA plug-in for CICS Explorer resource objects

This section describes the CIU_RESOURCE table. This table is used by the CICS IA plug-in to improve performance and combine resource tables.

CIU_RESOURCE

This CIU_RESOURCE table stores distinct information on all the resources collected by CICS IA, by region. It is reloaded every time the primary, dependency, or affinity tables are updated.

Table 61. The CIU_RESOURCE table

Field name	Type	Description
TYPE	CHAR(16)	Resource type
OBJECT	VAR CHAR(255)	Resource name
APPLID	CHAR(8)	CICS region applid

The structure of the Version objects

This section describes the CIU_VERSION table. This table is used by the CICS IA plug-in for CICS Explorer to synchronize the CICS IA plug-in version being used and the base IA database objects.

CIU_VERSION table

IBM Support will request the information in the CIU_VERSION table if you have any CICS IA plug-in issues. For more information see CICS IA Explorer Level in the Solving problems section.

The CIU_VERSION table stores APAR and release information for the CICS IA plug-in. When maintenance is applied to the base IA DB2 tables, or the CICS IA plug-in, reload the table using the sample job CIUVERLD.

Table 62. CIU_VERSION table

Field name	Type	Description
PRODID	CHAR(8)	Product ID"5967-J23"
DB_APAR_LEVEL	CHAR(8)	APAR level of the IA database
EXP_APAR_LEVEL	CHAR(8)	APAR level of the CICS IA plug-in
EXP_MIN_VER	CHAR(8)	Minimum level of the CICS IA plug-in required
EXP_LATEST_VER	CHAR(8)	Latest version of the CICS IA plug-in required
VER_DESC	CHAR(8)	Description of the latest CICS IA plug-in APAR
VER_CUST_DESC	CHAR(8)	Customer modifiable field that indicates action to take if a new CICS IA plug-in needs to be downloaded

The structure of the Command Flow table objects

This section describes the Command Flow base table that contains records for each command issued by the transaction(s). You can write your own SQL applications to query the table; these applications must use native SQL queries to do this.

CIU_CMDFLOW_DATA

This table stores information about every unique detail of the commands issued by the transaction.

Table 63. The CIU_CMDFLOW_DATA table

Column	Type	Description
CMDFLOW_ID	CHAR(08)	Name of the command flow trace.
TRACE_ID	CHAR(16)	Unique Command Flow run ID.
APPLID	CHAR(8)	CICS region APPLID.
SYSID	CHAR(4)	SYSID of local region.
TRANSID	CHAR(4)	CICS transaction ID.
TASKID	CHAR(4)	The transaction task ID, from which the command is invoked.
DISTRIBUTED_UOW	CHAR(27)	Distributed (Network) unit of work for task (the value of this field is the distributed UOW value recorded in the start of task).
CICS_UOW	CHAR(8)	CICS unit of work for task.
USERID	CHAR(8)	The CICS user ID of the transaction.
CONCURRENCY	CHAR(10)	Threadsafe. Quasirent. The value of this field is determined from the first program started for the transaction.
API	CHAR(8)	CICSAPI; OPENAPI. The value of this field is determined from the first program started for the transaction.
"PROGRAM"	CHAR(8)	The name of the CICS program invoking the command.
OFFSET	CHAR(8)	The offset of the command from the start of the program.
FUNCTION	CHAR(24)	Command name.
TYPE	CHAR(16)	Resource type.
FUNCTION_TYPE	CHAR(8)	The type of command, DB2, DLI, MQ, CICS, or the name of the TRUE exit if the type of command cannot be determined.
FUNCTION_ID	CHAR(4)	The function ID of the command.
FUNCTION_DESC	CHAR(32)	The description of the command (if known).
RESOURCE_NAME	CHAR(32)	The name of the resource the command is acting upon (if known).
TCBMODE	CHAR(2)	The TCB Mode value at the time the command is processed.
PREV_TCBMODE	CHAR(2)	The TCB Mode value recorded for the previous command. The value of this field is determined by the batch DB2 table upload program.
BEFORE_MODESWITCH	CHAR(1)	This field is determined by comparing the TCB mode of this command to the following command. The value is determined by the batch DB2 upload program. Y Last Command invoked before a TCB mode switch N Not the last command invoked before a TCB mode switch

Table 63. The CIU_CMDFLOW_DATA table (continued)

Column	Type	Description
AFTER_MODESWITCH	CHAR(1)	This field is determined by comparing the TCB mode of this command to the previous command. The value is determined by the batch DB2 upload program. Y Last Command invoked after a TCB mode switch N Not the last command invoked after a TCB mode switch
CICS_VERSION	CHAR(4)	The version of CICS from which the data is recorded.
CMD_TIME_LOCAL	TIMESTAMP	Time the command was issued, in the local time format.
CMD_EIBRESP	CHAR(08)	The response code is EIBRESP.
CMD_EIBRESP2	CHAR(08)	The response code is EIBRESP2.
CMD_EIDARG0_DATA	CHAR(56)	The Argument zero value of EXEC CICS command, in alphanumeric form.
CMD_USER_DAT1	CHAR(48)	User data 1 provided by the Command Flow User Exit program.
CMD_USER_DAT2	CHAR(48)	User data 2 provided by the Command Flow User Exit program.
CMD_USER_DAT3	CHAR(48)	User data 3 provided by the Command Flow User Exit program.

CIU_CMDFLOW_INDEX

This table contains all instances of the command flows that have been ran.

Table 64. The CIU_CMDFLOW_INDEX table

Column	Type	Description
OWNER_USERID	CHAR(8)	Name of USER collecting commands.
TRACE_ID	CHAR(16)	Unique Command Flow run ID.
CMDFLOW_ID	CHAR(08)	Name of the command flow trace.
COL_APPLID	CHAR(08)	Applid of the CICS region in which it was collected.
CMD_TIME_START	TIMESTAMP	Date and time of collection start, in the local time format.
CMD_TIME_END	TIMESTAMP	Date and time of collection end, in the local time format.
CMD_APPLID1	CHAR(08)	Applid of the CICS region 1 for this Command Flow run if specified.
CMD_APPLID2	CHAR(08)	Applid of the CICS region 2 for this Command Flow run if specified.
CMD_APPLID3	CHAR(08)	Applid of the CICS region 3 for this Command Flow run if specified.
CMD_APPLID4	CHAR(08)	Applid of the CICS region 4 for this Command Flow run if specified.
CMD_APPLID5	CHAR(08)	Applid of the CICS region 5 for this Command Flow run if specified.

Table 64. The CIU_CMDFLOW_INDEX table (continued)

Column	Type	Description
CMD_APPLID6	CHAR(08)	Applid of the CICS region 6 for this Command Flow run if specified.
CMD_APPLID7	CHAR(08)	Applid of the CICS region 7 for this Command Flow run if specified.
CMD_APPLID8	CHAR(08)	Applid of the CICS region 8 for this Command Flow run if specified.
CMD_APPLID9	CHAR(08)	Applid of the CICS region 9 for this Command Flow run if specified.
CMD_APPLID10	CHAR(08)	Applid of the CICS region 10 for this Command Flow run if specified.
CMD_APPLID11	CHAR(08)	Applid of the CICS region 11 for this Command Flow run if specified.
CMD_APPLID12	CHAR(08)	Applid of the CICS region 12 for this Command Flow run if specified.
CMD_APPLID13	CHAR(08)	Applid of the CICS region 13 for this Command Flow run if specified.
CMD_APPLID14	CHAR(08)	Applid of the CICS region 14 for this Command Flow run if specified.
CMD_APPLID15	CHAR(08)	Applid of the CICS region 15 for this Command Flow run if specified.
CMD_TRANID1	CHAR(04)	Transaction captured by the trace.
CMD_TRANID2	CHAR(04)	Transaction captured by the trace.
CMD_TRANID3	CHAR(04)	Transaction captured by the trace.
CMD_TRANID4	CHAR(04)	Transaction captured by the trace.
CMD_TRANID5	CHAR(04)	Transaction captured by the trace.
CMD_COUNT	INTEGER	Number of records captured during trace in this region.
CMD_APPLNAME	CHAR(8)	Application name.
CMD_USERID	CHAR(8)	Traced UserID.
CMD_TERMID	CHAR(4)	Traced TermID.
JOURNAL_NAME	CHAR(8)	Journal name.
JOURNAL_COPY_HLQ	CHAR(35)	Reserved.
USER_EXIT_NAME	CHAR(8)	The name of Command Flow User Exit program.

Type and Function mapping for monitored commands

Learn about the correspondence between resource type and command function for the monitored CICS commands.

The TYPE (resource type) and FUNCTION (command) columns have specific values defined by CICS IA at the time the data is loaded into DB2.

For a list of the possible combinations of TYPE and FUNCTION values in DB2 queries, see Table 107 on page 251.

For a list of the possible combinations of TYPE and FUNCTION values in IMS queries, see Table 108 on page 254.

For a list of the possible combinations of TYPE and FUNCTION values in MQ queries, see Table 109 on page 254.

Table 65. Type and Function mapping for monitored commands using the API ATOMServices CICS resource option flag

Resource type	Function	CICS command name
RECORD	BIF DIGEST	BIF DIGEST RECORD

Table 66. Type and Function mapping for monitored commands using the SPI ATOMServices CICS resource option flag

Resource type	Function	CICS command name
ATOMSERVICE	CREATE	CREATE ATOMSERVICE
	DISCARD	DISCARD ATOMSERVICE
	INQUIRE	INQUIRE ATOMSERVICE
	INQUIRE NEXT	INQUIRE ATOMSERVICE NEXT
	SET	SET ATOMSERVICE

Table 67. Type and Function mapping for monitored commands using the SPI BRFacility CICS resource option flag

Resource type	Function	CICS command name
BRFACIL	INQ NEXT	INQUIRE BRFACILITY NEXT
	INQUIRE	INQUIRE BRFACILITY
	SET	SET BRFACILITY

Table 68. Type and Function mapping for monitored commands using the SPI Bundles CICS resource option flag

Resource type	Function	CICS command name
BUNDLE	CREATE	CREATE BUNDLE
	DISCARD	DISCARD BUNDLE
	INQUIRE	INQUIRE BUNDLE
	INQUIRE NEXT	INQUIRE BUNDLE NEXT
	SET	SET BUNDLE
BUNDLEPART	INQUIRE	INQUIRE BUNDLEPART
	INQUIRE NEXT	INQUIRE BUNDLEPART NEXT

Table 69. Type and Function mapping for monitored commands using the SPI Corbaserver CICS resource option flag

Resource type	Function	CICS command name
CORBASRV	CREATE	CREATE CORBASERVER
	DISCARD	DISCARD CORBASERVER
	INQ NEXT	INQUIRE CORBASERVER NEXT
	INQUIRE	INQUIRE CORBASERVER
	PERFORM	PERFORM CORBASERVER
	SET	SET CORBASERVER

Table 70. Type and Function mapping for monitored commands using the API Counters CICS resource option flag

Resource type	Function	CICS command name
COUNTER	DEFINE	DEFINE COUNTER
	DEFINE	DEFINE DCOUNTER
	DELETE	DELETE COUNTER
	DELETE	DELETE DCOUNTER
	GET	GET COUNTER
	GET	GET DCOUNTER
	QUERY	QUERY COUNTER
	QUERY	QUERY DCOUNTER
	REWIND	REWIND COUNTER
	REWIND	REWIND DCOUNTER
	UPDATE	UPDATE COUNTER
	UPDATE	UPDATE DCOUNTER
POOL	DEF CTR	DEFINE COUNTER POOL
	DEF DCTR	DEFINE DCOUNTER POOL
	DEL CTR	DELETE COUNTER POOL
	DEL DCTR	DELETE DCOUNTER POOL
	GET CTR	GET COUNTER POOL
	GET DCTR	GET DCOUNTER POOL
	QRY CTR	QUERY COUNTER POOL
	QRY DCTR	QUERY DCOUNTER POOL
	REW CTR	REWIND COUNTER POOL
	REW DCTR	REWIND DCOUNTER POOL
	UPD CTR	UPDATE COUNTER POOL
	UPD DCTR	UPDATE DCOUNTER POOL

Table 71. Type and Function mapping for monitored commands using the SPI CSD CICS resource option flag

Resource type	Function	CICS command name
	CSDENDBRRSRCE	CSD ENDBRRSRCE
	CSDDISCONNECT	CSD DISCONNECT

Table 71. Type and Function mapping for monitored commands using the SPI CSD CICS resource option flag (continued)

Resource type	Function	CICS command name
GROUP	CSDADD	CSD ADD GROUP
	CSDALTER RESOURCE	CSD ALTER RESTYPE
	CSDCOPY	CSD COPY GROUP
	CSDCOPY GROUP TO	CSD COPY GROUP
	CSDCOPY RESOURCE IN	CSD COPY RESTYPE
	CSDCOPY RESOURCE AS	CSD COPY RESTYPE
	CSDCOPY RESOURCE TO	CSD COPY RESTYPE
	CSDDEFINE RESOURCE IN	CSD DEFINE RESTYPE
	CSDDELETE	CSD DELETE GROUP
	CSDDELETE RESOURCE	CSD DELETE RESTYPE
	CSDGETNEXTGROUP	CSD GETNEXTGROUP GROUP
	CSDGETNEXTRSRCE IN	CSD GETNEXTRSRCE RESTYPE
	CSDINQUIREGROUP	CSD INQUIREGROUP GROUP
	CSDINQUIREGROUP	CSD INQUIREGROUP GROUP LIST
	CSDINQIRERSRCE	CSD INQIRERSRCE RESTYPE
	CSDINSTALL	CSD INSTALL GROUP
	CSDINSTALL RESOURCE	CSD INSTALL RESTYPE
	CSDLOCK	CSD LOCK GROUP
	CSDREMOVE	CSD REMOVE GROUP
	CSDRENAME RESOURCE	CSD RENAME RESTYPE
	CSDSTARTBRGROUP	CSD STARTBRGROUP
	CSDSTARTBRRSRCE	CSD STARTBRRSRCE
	CSDENDBRGROUP	CSD ENDBRGROUP
	CSDUNLOCK	CSD UNLOCK GROUP
	CSDUSERDEFINE RESOURCE	CSD USERDEFINE RESTYPE

Table 71. Type and Function mapping for monitored commands using the SPI CSD CICS resource option flag (continued)

Resource type	Function	CICS command name
LIST	CSDADD GROUP TO	CSD ADD GROUP
	CSDAPPEND	CSD APPEND LIST
	CSDAPPEND TO	CSD APPEND LIST
	CSDDELETE	CSD DELETE LIST
	CSDENDBRLIST	CSD ENDBRLIST
	CSDGETNEXTLIST	CSD GETNEXTLIST LIST
	CSDINQUIREGROUP IN	CSD INQUIREGROUP GROUP LIST
	CSDINQUIRELIST	CSD INQUIRELIST LIST
	CSDINSTALL	CSD INSTALL LIST
	CSDLLOCK	CSD LOCK LIST
	CSDREMOVE GROUP	CSD REMOVE GROUP
	CSDSTARTBRLIST	CSD STARTBRLIST
	CSDUNLOCK	CSD UNLOCK LIST
RESOURCE	CSDALTER	CSD ALTER RESTYPE
	CSDCOPY	CSD COPY RESTYPE
	CSDDEFINE	CSD DEFINE RESTYPE
	CSDDELETE	CSD DELETE RESTYPE
	CSDGETNEXTRSRCE	CSD GETNEXTRSRCE RESTYPE
	CSDINQUIRERSRCE	CSD INQUIRERSRCE RESTYPE
	CSDINSTALL	CSD INSTALL RESTYPE
	CSDRENAME	CSD RENAME RESTYPE
	CSDUSERDEFINE	CSD USERDEFINE RESTYPE

Table 72. Type and Function mapping for monitored commands using the SPI DB2 CICS resource option flag

Resource type	Function	CICS command name
DB2ENTRY	CREATE	CREATE DB2ENTRY
	DISCARD	DISCARD DB2ENTRY
	INQ NEXT	INQUIRE DB2ENTRY NEXT
	INQUIRE	INQUIRE DB2ENTRY
	SET	SET DB2ENTRY
DB2TRAN	CREATE	CREATE DB2TRAN
	DISCARD	DISCARD DB2TRAN
	INQ NEXT	INQUIRE DB2TRAN NEXT
	INQUIRE	INQUIRE DB2TRAN
	SET	SET DB2TRAN

Table 73. Type and Function mapping for monitored commands using the SPI DJAR CICS resource option flag

Resource type	Function	CICS command name
DJAR	CREATE	CREATE DJAR
	DISCARD	DISCARD DJAR
	INQ NEXT	INQUIRE DJAR NEXT
	INQUIRE	INQUIRE DJAR
	PERFORM	PERFORM DJAR
JVMPROF	INQ NEXT	INQUIRE JVMPROFILE NEXT
	INQUIRE	INQUIRE JVMPROFILE

Table 74. Type and Function mapping for monitored commands using the API EVENT proc CICS resource option flag

Resource type	Function	CICS command name
EVENT	SIGNAL	SIGNAL EVENT

Table 75. Type and Function mapping for monitored commands using the SPI EVENT proc CICS resource option flag

Resource type	Function	CICS command name
EVENTBINDING	INQUIRE	INQUIRE EVENTBINDING
	INQUIRE NEXT	INQUIRE EVENTBINDING NEXT
	SET	SET EVENTBINDING
	INQUIRE CAPTURESPEC	INQUIRE CAPTURESPEC
EVENTPROCESS	INQUIRE	INQUIRE EVENTPROCESS
	SET	SET EVENTPROCESS
CAPTURESPEC	INQUIRE	INQUIRE CAPTURESPEC
	INQUIRE NEXT	INQUIRE CAPTURESPEC NEXT

Table 76. Type and Function mapping for monitored commands using the API Exits CICS resource option flag

Resource type	Function	CICS command name
EXIT	CALL	(Call to TRUE)

Table 77. Type and Function mapping for monitored commands using the SPI Exits CICS resource option flag

Resource type	Function	CICS command name
EXIT	DISABLE	DISABLE PROGRAM
	ENABLE	ENABLE PROGRAM
	EXTRACT	EXTRACT EXIT

Table 78. Type and Function mapping for monitored commands using the FEPI API CICS resource option flag

Resource type	Function	CICS command name
FEPI	EXTRACTF	FEPI EXTRACT FIELD
	EXTRACTS	FEPI EXTRACT STNS
	FREE	FEPI FREE
	ISSUE	FEPI ISSUE
	RECEIVE	FEPI RECEIVE DATASTREAM
	RECEIVE	FEPI RECEIVE FORMATTED
	REQTCKT	FEPI REQUEST PASSTICKET
	SEND	FEPI SEND DATASTREAM
	SEND	FEPI SEND FORMATTED
	START	FEPI START
FEPIPOOL	ALLOCATE	FEPI ALLOCATE POOL
	CONVERSE	FEPI CONVERSE DATASTREAM
	CONVERSE	FEPI CONVERSE FORMATTED
	EXTRACTC	FEPI EXTRACT CONV

Table 79. Type and Function mapping for monitored commands using the FEPI SPI CICS resource option flag

Resource type	Function	CICS command name
FEPINODE	INQ CONN	FEPI INQUIRE TARGET
	INQ NODE	FEPI INQUIRE NODE
	SET CONN	FEPI SET CONNECTION
	SET NODE	FEPI SET NODE
FEPIPOOL	ADD POOL	FEPI ADD POOL
	DEL POOL	FEPI DELETE POOL
	DISCPool	FEPI DISCARD POOL
	INQ POOL	FEPI INQUIRE POOL
	INSTPOOL	FEPI INSTALL POOL
	SET POOL	FEPI SET POOL
FEPISET	DISCPSET	FEPI DISCARD POOL
	INQ PSET	FEPI INQUIRE PROPERTYSET
	INSTPSET	FEPI INSTALL PROPERTYSET
FEPITGT	INQ TRGT	FEPI INQUIRE TARGET
	SET TRGT	FEPI SET CONNECTION

Table 80. Type and Function mapping for monitored commands using the SPI File CICS resource option flag

Resource type	Function	CICS command name
FILE	CREATE	CREATE FILE
	DISCARD	DISCARD FILE
	INQ NEXT	INQUIRE FILE NEXT
	INQUIRE	INQUIRE FILE
	SET	SET FILE

Table 81. Type and Function mapping for monitored commands using the API Files CICS resource option flag

Resource type	Function	CICS command name
FILE	DELETE	DELETE
	ENDBR	ENDBR
	READ	READ
	READ UPD	READ UPDATE
	READNEXT	READNEXT
	READPREV	READPREV
	RESETBR	RESETBR
	REWRITE	REWRITE
	STARTBR	STARTBR
	UNLOCK	UNLOCK
	WRITE	WRITE

Table 82. Type and Function mapping for monitored commands using the SPI IPCONN CICS resource option flag

Resource type	Function	CICS command name
IPCONN	CREATE	CREATE IPCONN
	DISCARD	DISCARD IPCONN
	INQUIRE	INQUIRE IPCONN
	SET	SET IPCONN

Table 83. Type and Function mapping for monitored commands using the API Journals CICS resource option flag

Resource type	Function	CICS command name
JOURNAL	WAIT	WAIT JOURNALNAME
	WAIT	WAIT JOURNALNUM
	WRITE	WRITE JOURNALNAME
	WRITE	WRITE JOURNALNUM

Table 84. Type and Function mapping for monitored commands using the SPI Journals CICS resource option flag

Resource type	Function	CICS command name
JOURNAL	DISCARD	DISCARD JOURNALNAME
	INQ NEXT	INQUIRE JOURNALNAME NEXT
	INQ NEXT	INQUIRE JOURNALNUM NEXT
	INQUIRE	INQUIRE JOURNALNAME
	INQUIRE	INQUIRE JOURNALNUM
	SET	SET JOURNALNAME
	SET	SET JOURNALNUM

Table 85. Type and Function mapping for monitored commands using the SPI JVMServer CICS resource option flag

Resource type	Function	CICS command name
JVMSERVER	CREATE	CREATE JVMSERVER
	INQUIRE	INQUIRE JVMSERVER
	INQUIRE NEXT	INQUIRE JVMSERVER NEXT
	DISCARD	DISCARD JVMSERVER
	SET	SET JVMSERVER

Table 86. Type and Function mapping for monitored commands using the SPI Library CICS resource option flag

Resource type	Function	CICS command name
LIBRARY	CREATE	CREATE LIBRARY
	DISCARD	DISCARD LIBRARY
	INQUIRE	INQUIRE LIBRARY
	SET	SET LIBRARY

Table 87. Type and Function mapping for monitored commands using the SPI MQCONN CICS resource option flag

Resource type	Function	CICS command name
MQCONN	CREATE	CREATE MQCONN
	INQUIRE	INQUIRE MQCONN
	DISCARD	DISCARD MQCONN
	SET	SET MQCONN
MQINI	INQUIRE	INQUIRE MQINI
	DISCARD	DISCARD MQINI

Table 88. Type and Function mapping for monitored commands using the API Others CICS resource option flag

Resource type	Function	CICS command name
	ADDRESS	ADDRESS
	ALLOCATE	ALLOCATE
DOCTEMP	CREATE	CREATE DOCTEMPLATE
	DISCARD	DISCARD DOCTEMPLATE
	INQ NEXT	INQUIRE DOCTEMPLATE NEXT
	INQUIRE	INQUIRE DOCTEMPLATE
HANDLE	PUSH	PUSH HANDLE
	POP	POP HANDLE
STORAGE	FREEMAIN	FREEMAIN
	GETMAIN	GETMAIN
STORSHR	GETMAIN	GETMAIN SHARED
UOW	ROLLBACK	SYNCPOINT ROLLBACK
	SYNC	SYNCPOINT

Table 89. Type and Function mapping for monitored commands using the API Presentation CICS resource option flag

Resource type	Function	CICS command name
	RECEIVE	RECEIVE
	ROUTE	ROUTE
	SIGNOFF	SIGNON
	SIGNON	SIGNOFF
ABEND	ISSUE	ISSUE ABEND
CONFRMTN	ISSUE	ISSUE CONFIRMATION
COPY	ISSUE	ISSUE COPY
DISCONNT	ISSUE	ISSUE DISCONNECT
ERROR	ISSUE	ISSUE ERROR
MAP	PURGE	PURGE MESSAGE
	RECEIVE	RECEIVE MAP
	SEND	SEND MAP
MAPSET	RECV MAP	RECEIVE MAP MAPSET
	SEND MAP	SEND MAP MAPSET
PASS	ISSUE	ISSUE PASS
PROCESS	EXTRACT	EXTRACT PROCESS
RESET	ISSUE	ISSUE RESET
SIGNAL	ISSUE	ISSUE SIGNAL
TERMINAL	WAIT	WAIT TERMINAL
TEXT	SEND	SEND TEXT

Table 90. Type and Function mapping for monitored commands using the API Presentation or the API DTP CICS resource option flag

Resource type	Function	CICS command name
	CONVERSE	CONVERSE
	FREE	FREE
	SEND	SEND
PROCESS	CONNECT	CONNECT PROCESS

Table 91. Type and Function mapping for monitored commands using the API Presentation or the API Others CICS resource option flag

Resource type	Function	CICS command name
	ASSIGN	ASSIGN

Table 92. Type and Function mapping for monitored commands using the SPI Programs CICS resource option flag

Resource type	Function	CICS command name
PROGRAM	CREATE	CREATE PROGRAM
	DISCARD	DISCARD PROGRAM
	INQ NEXT	INQUIRE PROGRAM NEXT
	INQUIRE	INQUIRE PROGRAM
	SET	SET PROGRAM

Table 93. Type and Function mapping for monitored commands using the API Programs CICS resource option flag

Resource type	Function	CICS command name
CHANNEL	DEL CNTR	DELETE CONTAINER CHANNEL
	GET CNTR	GET CONTAINER CHANNEL
	LINK	LINK PROGRAM CHANNEL
	MOV CNTR	MOVE CONTAINER CHANNEL
	PUT CNTR	PUT CONTAINER CHANNEL
	RETURN	RETURN CHANNEL
	XCTL	XCTL PROGRAM CHANNEL
CONTAINR	DELETE	DELETE CONTAINER
	GET	GET CONTAINER
	MOVE	MOVE CONTAINER
	PUT	PUT CONTAINER
PROGRAM	CALL	Dynamic program call
	HANDABND	HANDLE ABEND
	LINK	LINK
	LOAD	LOAD
	XCTL	XCTL

Table 94. Type and Function mapping for monitored commands using the API Task Control CICS resource option flag

Resource type	Function	CICS command name
ENQNAME	DEQ	DEQ
	DEQSYS	DEQ (scope is sysplex-wide)
	ENQ	ENQ
	ENQSYS	ENQ (scope is sysplex-wide)

Table 95. Type and Function mapping for monitored commands using the SPI TCPIPService CICS resource option flag

Resource type	Function	CICS command name
TCPIPSRV	CREATE	CREATE TCPIPService
	DISCARD	DISCARD TCPIPService
	INQ NEXT	INQUIRE TCPIPService NEXT
	INQUIRE	INQUIRE TCPIPService
	SET	SET TCPIPService

Table 96. Type and Function mapping for monitored commands using the API TD Queues CICS resource option flag

Resource type	Function	CICS command name
TD	DELETEQ	DELETEQ TD
	READQ	READQ TD
	WRITEQ	WRITEQ TD

Table 97. Type and Function mapping for monitored commands using the SPI Temp Storage CICS resource option flag

Resource type	Function	CICS command name
TS	INQ NEXT	INQUIRE TSQNAME NEXT
	INQ NEXT	INQUIRE TSQUEUE NEXT
	INQUIRE	INQUIRE TSQNAME
	INQUIRE	INQUIRE TSQUEUE
	SET	SET TSQNAME
	SET	SET TSQUEUE
TSMODEL	CREATE	CREATE TSMODEL
	DISCARD	DISCARD TSMODEL
	INQUIRE	INQUIRE TSMODEL
TSPPOOL	INQUIRE	INQUIRE TSPPOOL

Table 98. Type and Function mapping for monitored commands using the API Transactions CICS resource option flag

Resource type	Function	CICS command name
TRANSID	RETURN	RETURN
	START	START
	START	START ATTACH
	START	START BREXIT
	STARTREQ	START REQID

Table 99. Type and Function mapping for monitored commands using the SPI Transactions CICS resource option flag

Resource type	Function	CICS command name
TRANSID	CREATE	CREATE TRANSACTION
	DISCARD	DISCARD TRANSACTION
	INQ NEXT	INQUIRE TRANSACTION NEXT
	INQUIRE	INQUIRE TRANSACTION
	SET	SET TRANSACTION

Table 100. Type and Function mapping for monitored commands using the SPI Transient Data CICS resource option flag

Resource type	Function	CICS command name
TD	CREATE	CREATE TDQUEUE
	DISCARD	DISCARD TDQUEUE
	INQ NEXT	INQUIRE TDQUEUE NEXT
	INQUIRE	INQUIRE TDQUEUE
	SET	SET TDQUEUE

Table 101. Type and Function mapping for monitored commands using the API TS Queues CICS resource option flag

Resource type	Function	CICS command name
CHANNEL	START	START CHANNEL

Table 101. Type and Function mapping for monitored commands using the API TS Queues CICS resource option flag (continued)

Resource type	Function	CICS command name
TS	DELETEQ	DELETEQ TS
	READQ	READQ TS
	WRITEQ	WRITEQ TS
TSAU	DELETEQ	DELETEQ TS (auxiliary storage)
	READQ	READQ TS (auxiliary storage)
	WRITEQ	WRITEQ TS (auxiliary storage)
TSSHR	DELETEQ	DELETEQ TS (shared)
	READQ	READQ TS (shared)
	WRITEQ	WRITEQ TS (shared)

Table 102. Type and Function mapping for monitored commands using the API Web Services CICS resource option flag

Resource type	Function	CICS command name
SERVICE	INVOKE	INVOKE SERVICE
WEB	ENDBR	WEB ENDBROWSE
	EXTRACT	WEB EXTRACT
	READ	WEB READ
	READNEXT	WEB READNEXT
	RECEIVE	WEB RECEIVE
	RETRIEVE	WEB RETRIEVE
	SEND	WEB SEND
	STARTBR	WEB STARTBROWSE
	WRITE	WEB WRITE HTTPHEADER
WEBSRV	CALL	A web service call into CICS
	INVOKE	INVOKE WEBSERVICE

Table 103. Type and Function mapping for monitored commands using the SPI Web Services CICS resource option flag

Resource type	Function	CICS command name
PIPELINE	CREATE	CREATE PIPELINE
	DISCARD	DISCARD PIPELINE
	INQ NEXT	INQUIRE PIPELINE NEXT
	INQUIRE	INQUIRE PIPELINE
	PERFORM	PERFORM PIPELINE
	SET	SET PIPELINE
URIMAP	CREATE	CREATE URIMAP
	DISCARD	DISCARD URIMAP
	INQ NEXT	INQUIRE URIMAP NEXT
	INQUIRE	INQUIRE URIMAP
	SET	SET URIMAP

Table 103. Type and Function mapping for monitored commands using the SPI Web Services CICS resource option flag (continued)

Resource type	Function	CICS command name
WEBSRV	CREATE	CREATE WEBSERVICE
	DISCARD	DISCARD WEBSERVICE
	INQ NEXT	INQUIRE WEBSERVICE NEXT
	INQUIRE	INQUIRE WEBSERVICE
	SET	SET WEBSERVICE

Table 104. Type and Function mapping for monitored commands using the API WSAddressing CICS resource option flag

Resource type	Function	CICS command name
CHANNEL	BUILD WSACONTEXT	WSACONTEXT BUILD
	GET WSACONTEXT	WSACONTEXT GET
	DELETE WSACONTEXT	WSACONTEXT DELETE
WSACONTEXT	BUILD	WSACONTEXT BUILD
	GET	WSACONTEXT GET
	DELETE	WSACONTEXT DELETE
WSAEPR	CREATE	WSAEPR CREATE

Table 105. Type and Function mapping for monitored commands using the API XMLTransform CICS resource option flag

Resource type	Function	CICS command name
DATATOXML	TRANSFORM	TRANSFORM XFORMTYPE(DATATOXML)
XMLTODATA	TRANSFORM	TRANSFORM XFORMTYPE(XMLTODATA)

Table 106. Type and Function mapping for monitored commands using the SPI XMLTransform CICS resource option flag

Resource type	Function	CICS command name
XMLTRANSFORM	INQUIRE	INQUIRE XMLTRANSFORM
	INQUIRE NEXT	INQUIRE XMLTRANSFORM NEXT
	SET	SET XMLTRANSFORM

Table 107. The possible combinations of TYPE and FUNCTION values in DB2 queries

RESOURCE TYPE	FUNCTION
Dynamic	EXECUTE IMMEDIATE
CURSOR	OPEN
	FETCH
	CLOSE
	ALLOCATE CURSOR

Table 107. The possible combinations of TYPE and FUNCTION values in DB2 queries (continued)

RESOURCE TYPE	FUNCTION
TABLE	SELECT
	INSERT
	DELETE
	UPDATE
	RENAME TABLE
	CREATE TABLE
	ALTER TABLE
	DROP TABLE
VIEW	CREATE VIEW
	DROP VIEW
ALIAS	CREATE ALIAS
	DROP ALIAS
SYNONYM	CREATE SYNONYM
	DROP SYNONYM
PACKAGE	DROP PACKAGE/PROGRAM
STATEMENT	PREPARE
	EXECUTE
	DESCRIBE
INDEX	CREATE INDEX
	DROP INDEX
	ALTER INDEX
STOGROUP	CREATE STOGROUP
	DROP STOGROUP
	ALTER STOGROUP
TABLESPACE	CREATE TABLESPACE
	DROP TABLESPACE
	ALTER TABLESPACE
DATABASE	CREATE DATABASE
	DROP DATABASE
	ALTER DATABASE

Table 107. The possible combinations of TYPE and FUNCTION values in DB2 queries (continued)

RESOURCE TYPE	FUNCTION
None	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 LOACATION && HV
	RELEASE CURRENT
	RELEASE ALL
	RELEASE ALL SQL
	RELEASE ALL PRIVATE
	SET CURRENT RULESS
	CALL STATEMENT
	DESCRIBE PROCEDURE
	ASSOCIATE LOCATORS
	FETCH ALLOC CURSOR
	CLOSE ALLOC CURSOR
DESCRIBE ALLOC CURSOR	
DESCRIBE INPUT	
SET SPECIAL REGISTER	

Table 108. The possible combinations of TYPE and FUNCTION values in IMS queries

RESOURCE TYPE	FUNCTION
PCB	DELETE
	GET NEXT
	GET NEXT INP
	GET UNIQUE
	INSERT
	REPLACE
PSB	SCHEDULE

Table 109. The possible combinations of TYPE and FUNCTION values in MQ queries

RESOURCE TYPE	FUNCTION
BUFFER	CONVERT
CALLBACK	MANAGE
	CONTROL
MESSAGE HANDLE	CREATE
	CONVERT
	DELETE
MESSAGE PROPERTY	SET
	INQUIRE
	DELETE
QUEUE	CLOSE
	GET
	OPEN
	PUT
	PUT1
	INQUIRE
SUBSCRIPTION	REGISTER
	REQUEST
	STATUS

Table 110. The possible combinations of TYPE and FUNCTION values in Natural queries

RESOURCE TYPE	FUNCTION
ADABAS	CALL
PROGRAM	CALL

Appendix D. Messages and codes

This section describes the messages that the Collector, Query interface, Dependency Reporter, Affinities Reporter, Load Module Scanner, CSECT Scanner, and Builder can issue, and the transaction abend codes that the Collector can produce.

The messages and abend codes are described in alphabetic order, starting on page.

As an aid to problem determination, this section also lists the meaning for each possible value of the call parameters that are included in the error messages issued if an error occurs on a call to the:

- Collector table manager, CIUTABM. See “Collector table manager diagnostics” on page 314.
- Collector CINB request queue manager, CIUCINP. See page “Collector CINB request queue manager diagnostics” on page 316.
- CICS IA date formatter, CIUCINDT. See page “Date formatter diagnostics” on page 316.

Contacting IBM Support

Information on IBM support policy can be found on our Web site.

Follow the Support link in the left-hand column at ibm.com/software/ts/cics/

Messages that CICS IA can issue

CIU1000E CIUMSGE Language Module Not Found

Explanation: The English Language message module has not been found.

System action: None.

User response: Check the CICS IA load library @hlq.SCIULODE is in the CICS DFHRPL concatenation.

Module: CIUIVPC

Destination

Terminal end user and CINT TD Queue.

CIU1001I Begin CICS IA IVP for TS version: *CICS-version-number*

Explanation: The CICS IA installation verification program (IVP) has started on this CICS TS region. *CICS-version-number* is the CICS version number; for example, 2.3.

System action: The IVP checks that CICS IA has been installed correctly.

User response: Check the CICS IA load library @hlq.SCIULODE is in the CICS DFHRPL concatenation.

Module: CIUIVPC

Destination

Terminal end user and CINT TD queue.

CIU1002I Installation verification ended successfully

Explanation: CICS IA has been installed correctly on this region.

System action: None.

User response: None.

Module: CIUIVPC

Destination

Terminal end user and CINT TD queue.

CIU1003E Transaction verification failed, transaction ID: *transaction-name*

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, transaction *transaction-name*, has not been defined to CICS correctly, or is not available.

System action: The CICS IA IVP continues to run, to check whether other software objects of CICS IA have been installed correctly.

CIU1004E • CIU1008I

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG13, is included in the startup group list.
2. Use the CEDA transaction to verify that the required transaction, *transaction-name*, is included in the CIUnnG13 resource group.
3. If transaction *transaction-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU1004E Program verification failed, program ID:
program-name

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, file *program-name*, has not been defined to CICS correctly, or is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG13, is included in the startup group list.
2. Use the CEDA transaction to verify that the required program, *program-name*, is included in the CIUnnG13 resource group.
3. Check that the CICS IA load library has been added to the DFHRPL list at CICS startup.
4. If program *program-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU1005E VSAM file verification failed, file ID:
file-name

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, file *file-name*, has not been defined to CICS correctly, or is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG13, is included in the startup group list.

2. Use the CEDA transaction to verify that the required file, *file-name*, is included in the CIUnnG13 resource group.
3. If file *file-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue

CIU1006I Transaction verified, transaction ID:
transaction-name

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, transaction *transaction-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1007I Program verified, program ID:
program-name

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, program *program-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1008I VSAM file verified, file ID: *file-name*

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, VSAM file *file-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1009E **Verification unsuccessful — highest return code:** *return-code*

Explanation: The CICS IA installation verification program (IVP) has found that CICS IA has not been installed correctly on this region.

System action: None.

User response:

1. Investigate the cause of the problem by examining the CICS IA messages in the CICS system log. CICS IA messages are in the range CIU1001 through CIU1013.
2. Locate any missing resources identified by the IVP. Ensure that all the resources required by CICS IA are correctly defined to CICS and are available.
3. If you cannot locate or restore a missing resource, contact the IBM Software Support Center (ISC).
4. Re-run the IVP until it confirms that CICS IA has been installed correctly.

Module: CIUIVPC

Destination

Terminal end user and CINT TD queue.

CIU1010I **TD queue verified, TD queue ID:** *TDqueue-name*

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, TD queue *TDqueue-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1011E **TD queue verification failed, TD queue ID:** *TDqueue-name*

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, transient data queue *TDqueue-name*, has not been defined to CICS correctly, or is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG13, is included in the startup group list.
2. Use the CEDA transaction to verify that the required TD queue, *TDqueue-name*, is included in the CIUnnG13 resource group.

3. If TD queue *TDqueue-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU1012I **DB2ENTRY verified, DB2ENTRY ID:** *DB2entry-name*

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, DB2ENTRY *DB2entry-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1013E **DB2ENTRY verification failed, DB2ENTRY ID:** *DB2ENTRY-name*

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, DB2 entry *DB2ENTRY-name*, has not been defined to CICS correctly, or is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG13 is included in the startup group list.
2. Use the CEDA transaction to verify that the required DB2 entry *DB2ENTRY-name*, is included in the CIUnnG13 resource group.
3. If DB2 entry *DB2ENTRY-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU1014I **DB2TRAN verified, DB2TRAN ID:** *DB2TRAN-name*

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, DB2TRAN *DB2TRAN-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

CIU1015E • CIU2103W

Module: CIUIVPC

Destination

CINT TD queue

CIU1015E **DB2TRAN verification failed,
DB2TRAN ID: DB2TRAN-name**

Explanation: The CICS IA installation verification program (IVP) has found that a required resource, DB2TRAN DB2TRAN-name, has not been defined correctly or is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response: Use the CEDA transaction to verify that the required CICS IA resource group, CIUnnG22 is included in the startup group list. Use the CEDA transaction to verify that the required DB2TRAN DB2TRAN-name, is included in the CIUnnG22 resource group. If DB2TRAN DB2TRAN-name is missing, contact your IBM Software Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU1016 **JOURNALMODEL verified, model ID:
model-name**

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, journal model *model-name*, has been defined to CICS correctly, and is available.

System action: None.

User response: None.

Module: CIUIVPC

Destination

CINT TD queue.

CIU1017 **JOURNALMODEL verification failed,
model ID: model-name**

Explanation: The CICS IA installation verification program (IVP) has verified that a required resource, journal model *model-name*, has not been defined to CICS correctly, and is not available.

System action: The CICS IA IVP continues to run, to check whether other software elements of CICS IA have been installed correctly.

User response:

1. Use the CEDA transaction to verify that the required CICS IA resource group is included in the startup group list.

2. Use the CEDA transaction to verify that the required file, *model-name*, is included in the resource group.
3. If the file *model-name* is missing, contact your IBM Software Support Center (ISC).

Module: CIUIVPC

Destination

CINT TD queue.

CIU2101W **CINT already in use by user *userid***

Explanation: The CINT transaction is already being used when an attempt is made to start another CINT transaction.

System action: The second CINT transaction is terminated, as only one instance of CINT is permitted.

User response: Check where CINT is currently in use. Only one user should be attempting to run the Collector at a given time.

Module: CIUA000C

Destination

Terminal end user or CINT TD queue.

CIU2102W **Collector is not *state***

Explanation: An attempt was made to Start, Stop, Pause, or Continue the Collector from CINT. However, the Collector was not currently in an appropriate *state* to make the change.

System action: The Collector state is not changed.

User response: Check why the Collector is currently in that state.

Module: CIUA000C, CIUA100C

Destination

Terminal end user or CINT TD queue.

CIU2103W **Collector is already stopped**

Explanation: An attempt was made to Stop the Collector from CINT when the Collector was already STOPPED.

System action: The Collector state is not changed.

User response: Check why the Collector is currently in that state.

Module: CIUA000C

Destination

Terminal end user or CINT TD queue.

CIU2104W Invalid key was pressed

| **Explanation:** The terminal operator has pressed a
 | function key in response to a screen displayed by the
 | CINT or CINC transaction, but the function key was
 | not valid for that screen.

System action: The function key is ignored.

User response: Use the correct function key.

| **Module:** CIUA00HC, CIUA000C, CIUA100C,
 | CIUA150C, CIUA200C, CIUA240C, CIUA250C,
 | CIUA260C, CIUA300C, CIUA400C, CIUA410C,
 | CIUA420C, CIUA440C, CIUA900C, CIUACM00,
 | CIUACM10, CIUACM20, CIUACM30, CIUACM40.

Destination

Terminal end user.

CIU2105I CINT session has ended

Explanation: The transaction CINT has ended.

System action: The state of the Collector is unchanged.

User response: None.

Module: CIUA000C

Destination

Terminal end user and CINT TD queue.

CIU2106W Options must be Y(Yes) or N(No)

Explanation: The only valid values for this CINT operation option are 'Y' and 'N'.

System action: The CINT operation options will not be changed unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUIVPC

Destination

CIUA240C, CIUA250C, CIUA260C, CIUA270C,
 CIUA280C, CIUA300C

CIU2107W Size must be integer (10 to 2000 Mb)

Explanation: The only valid value for the Collector data space size is an integer in the range 10 to 2000, in megabytes.

System action: The CINT operation options will not be changed unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal end user.

CIU2108W Select a valid date/time format

Explanation: An invalid date/time format has been entered.

System action: None.

User response: Select a valid date/time format. See HELP panel for valid formats.

Module: CIUA300C

Destination

Terminal end user.

CIU2109W Select a valid date/time separator

Explanation: An invalid date/time separator has been entered.

System action: None.

User response: Select a valid date/time separator. See HELP panel for valid formats.

Module: CIUA300C

Destination

Terminal end user.

CIU2110I No amendments were entered

| **Explanation:** The Enter key was pressed, but no
 | changes had been input.

System action: No options are changed.

User response: Enter any changes and then press Enter again.

| **Module:** CIUA240C, CIUA250C, CIUA260C,
 | CIUA300C, CIUACM10, CIUACM20.

Destination

Terminal end user.

CIU2111I CINT options are updated

Explanation: The CINT transaction has successfully amended the operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA240C, CIUA250C, CIUA260C,
 CIUA300C

Destination

Terminal end user and CINT TD queue.

CIU2113W Options must be Y(Yes), N(No) or blank(default)

Explanation: The only valid values for this CINT operation option are 'Y', 'N', and ' '.

System action: The CINT operation options will not be changed unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUA240C, CIUA250C, CIUA260C, CIUA270C, CIUA280C

Destination

Terminal end user.

CIU2114I Records are restored

Explanation: The Collector is being started with the restore data option set to Y. Records from the previous Collector run are retained on the dependency data file, CIUINT1.

Records for those dependency command types that are being detected on this Collector run were found on the files and were read into the data space.

System action: The Collector is started with any records from a previous run retained on the dependency data file and read into the data space.

User response: None.

Module: CIUA110C

Destination

Terminal end user.

CIU2115I Dependency files are emptied

Explanation: The Collector is being started with the restore data option set to N. All existing records were deleted from the dependency data file, CIUINT1.

System action: The Collector is started with an empty dependency data file.

User response: None.

Module: CIUA110C

Destination

Terminal end user and CINT TD queue.

CIU2116W Dataspace too large - no storage available

Explanation: The Collector is being started and it received a response from the table manager, CIUTABM, that it was unable to obtain the amount of storage requested for the MVS data space, because the MVS Real Storage Manager does not have enough resources.

System action: The Collector start is aborted and the Collector is stopped.

User response: Decrease the data space size using the CINT operation options.

Module: CIUA110C

Destination

Terminal end user.

CIU2117W Dataspace too large - IEFUSI limit reached

Explanation: The Collector is being started and it received a response from the table manager, CIUTABM, that it was unable to obtain the amount of storage requested for the MVS data space, because MVS exit IEFUSI has imposed a limit on address space size.

System action: The Collector start is aborted and the Collector is stopped.

User response: Decrease the data space size using the CINT operation options or else ask the MVS system programmer to increase the IEFUSI limit.

Module: CIUA110C

Destination

Terminal end user.

CIU2118I No records were restored

Explanation: The Collector is being started with the restore data option set to Y. Records from the previous Collector run are retained on the dependency data file, CIUINT1

No records for those dependency command types that are being detected on this Collector run were found on the files, so none were read into the data space.

System action: The Collector is started with any records from a previous run retained on the dependency data file.

User response: None.

Module: CIUA110C

Destination

Terminal end user.

CIU2119I CICS is terminating

| **Explanation:** The CINT or CINC transaction has
| detected that CICS is terminating.

System action: CICS IA is stopped.

User response: None.

| **Module:** CIUA000C, CIUACM10,

Destination

Terminal end user and CINT TD queue.

CIU2120I **Press Enter to confirm Start with data restore or PF12 to cancel**

Explanation: Confirmation is required that the Collector is to be started with the restore data option set to Y.

System action: If Enter is pressed, the Collector is started with dependency data retained on the dependency data file and read into the data space. Otherwise, F12 must be pressed to cancel the collection operation.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2121I **Press Enter to confirm Start without restore or PF12 to cancel**

Explanation: Confirmation is required that the Collector is to be started with the restore data option set to N.

System action: If Enter is pressed, the Collector is started and all of the data on the dependency data file deleted. Otherwise, F12 must be pressed to cancel the collection operation.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2122I **Press Enter to confirm Stop or PF12 to cancel**

Explanation: Confirmation is required that the Collector is to be stopped. The dependency data in the data space will be saved to the dependency data file.

System action: If Enter is pressed, the Collector is stopped and any changes made to the dependency data in the data space since the last save are saved to the dependency data file. Otherwise, F12 must be pressed to cancel the stop request.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2123I **Enter to confirm Start ALL or PF12 to cancel**

Explanation: Confirmation is required that the Collector is to be started in all regions.

System action: If Enter is pressed then the collector will be started in all regions where it can be started. Otherwise, PF12 must be pressed and the Collector will not be started.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2124I **Enter to confirm Stop ALL or PF12 to cancel**

Explanation: Confirmation is required that the Collector is to be stopped in all regions.

System action: If Enter is pressed then the collector will be stopped in all regions where it can be stopped. Otherwise, PF12 must be pressed and the Collector will not be stopped.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2125E **A CINT action must be supplied**

Explanation: The transid CINT is being entered at a console device. It is mandatory to supply a Collector action with CINT at a console device.

System action: The CINT transaction is not initiated.

User response: Ensure that an action, one of START, STOP, PAUSE, CONTINUE, is entered after the transid CINT.

Module: CIUA000C

Destination

Console.

CIU2126I **No actions to take**

Explanation: The Enter key was pressed but no action code has been entered, or an action code for ALL regions was entered, but the action could not be applied to any regions.

System action: None. There was nothing to do.

User response: If you want something to happen then key in an action code before pressing Enter.

Module: CIUA100C

Destination

Terminal end user.

CIU2127W Transaction ID prefix is invalid

Explanation: The transid prefix input on screen CINT02 is invalid. The input was rejected because it contained an embedded blank space character.

System action: The CINT options will not be changed unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal end user.

CIU2128W Language code is invalid

Explanation: The language code input on the screen is invalid.

System action: The CINT option will not be changed unless all of the input is correct.

User response: Enter a valid language code.

Module: CIUA300C

Destination

Terminal end user.

CIU2129W Control file CIUCNTL not open in the FOR

Explanation: CICS IA encountered an error while trying to open the CIUCNTL control file in the file-owning region.

| **System action:** The CINT or CINC transaction is stopped.

User response: Contact your CICS system support.

| **Module:** CIUA000C, CIUACM10, CIUACM60,
| CIUACM70.

Destination

CINT TD queue and terminal end user.

CIU2130W Dependency file CIUINTh not open in the FOR

Explanation: CICS IA encountered an error while trying to open the dependency file CIUINTh in the file-owning region.

System action: The CINT transaction is stopped.

User response: Contact your CICS system support.

Module: CIUA000C

Destination

INT TD queue and terminal end user.

CIU2131W Control file CIUCNTL is not available

Explanation: The CICS IA Control file, CIUCNTL, is disabled.

System action: None.

User response: Contact your CICS system support person.

| **Module:** CIUA000C, CIUA400C, CIUA420C,
| CIUA440C, CIUACM10, CIUACM20, CIUACM31,
| CIUACM60, CIUACM70, CIUATM03.

Destination

Terminal end user.

CIU2132W Dependency file CIUINT1 is not available

Explanation: The CICS IA dependency file, CIUINT1, is disabled.

System action: None.

User response: Contact your CICS system support person.

Module: CIUA000C

Destination

Terminal end user.

CIU2133W Error opening Control file CIUCNTL

Explanation: CICS IA encountered an error while trying to open the CIUCNTL control file.

System action: None.

User response: Contact your CICS system support person.

| **Module:** CIUA000C, CIUA400C, CIUA420C,
| CIUA440C, CIUACM10, CIUACM20, CIUACM31,
| CIUACM60, CIUACM70, CIUATM03.

Destination

Terminal end user.

CIU2134W Error opening Dependency file CIUINT1

Explanation: CICS IA encountered an error while trying to open the CIUINT1 dependency file.

System action: None.

User response: Contact your CICS system support person.

Module: CIUA000C

Destination

Terminal end user.

CIU2135S CICS command data failed RESP=eibresp
RESP2=eibresp2 RCODE=eibrcode

Explanation: Transaction CINT, CINB, or CINC received an invalid response when issuing EXEC CICS command. The response is in *eibresp*, *eibresp2* and *eibrcode*. Other *data*, if present, might give the object operated on by the *command*.

System action: The CINT, CINB, or CINC transaction continues. A message is sent to the terminal. The requested action fails.

User response: For further details of the exception *eibresp* refer to the *command* in the *CICS Application Programming Reference* manual or the *CICS System Programming Reference* manual.

For further information on how to determine system problems refer to the *CICS Problem Determination Guide*.

Module: CIUA000C, CIUACM10, CIUACM20, CIUACM30, CIUACM31, CIUACM40, CIUACM60, CIUACM70.

Destination

Terminal end user and CINT TD queue.

CIU2136W Exclude list name is invalid

Explanation: The name of the program exclude list or the transaction exclude list that has been input on the screen is invalid.

System action: The CINT options will not be changed unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal end user.

CIU2137I Affinity files are emptied

Explanation: The Collector is being started with the restore data option set to N. All existing records were deleted from the affinity data files.

System action: The Collector is started with an empty affinity data file.

User response: None.

Module: CIUA110C

Destination

Terminal end user and CINT TD queue.

CIU2138W Options must be Y(Yes), N(No), or D(Detail)

Explanation: The valid values for this operation are Y, N, and D.

System action: The CINT operation options will not change until all of the input is correct.

User response: Correct the invalid input.

Module: CIUA240C

Destination

Terminal end user

CIU2139W Options must be Y(Yes), N(No), D(Detail) or blank(default)

Explanation: The valid values for this operation are Y, N, D, and blank.

System action: The CINT operation options will not change until all of the input is correct.

User response: Correct the invalid input.

Module: CIUA240C

Destination

Terminal end user

CIU2140W Value must be between 2 and 9999, or 1 for no updates

Explanation: Enter the value in thousands between 2 and 9999. This relates to the number of records to be updated before CICS IA triggers the save transaction. A value of 1 indicates that no saves will be triggered.

System action: None

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal End User

CIU2141W Options must be A (Affinity), I (Interdependency), or B (Both)

Explanation: The valid values for this option are A, I, or B.

System action: The value changes only if the input is correct.

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal end user.

CIU2142W Options must be A (Affinity), I (Interdependency), B (Both), or blank

Explanation: The valid values for this option are A, I, B, or blank.

System action: The value changes only if the input is correct.

User response: Correct the invalid input.

Module: CIUA260C

Destination

Terminal end user.

CIU2143W Options must be Y (Yes), N (No), A (Aff.), I (Dep.), or B (Both)

Explanation: The valid values for this option are Y, N, A, I, or B.

System action: The value changes only if the input is correct.

User response: Correct the invalid input.

Module: CIUA280C

Destination

Terminal end user.

CIU2144W Options must be Y (Yes), N (No), A (Aff.), I (Dep.), B (Both), or blank

Explanation: The valid values for this option are Y, N, A, I, B, or blank.

System action: The value changes only if the input is correct.

User response: Correct the invalid input.

Module: CIUA280C

Destination

Terminal end user.

CIU2146I Press Enter to confirm REFRESH or PF12 to cancel

Explanation: Confirmation that the Collector is to use the updated CICS IA options is required.

System action: If Enter is pressed, the Collector performs the following steps:

1. Reads the options from the control file.
2. Disables collection of data that is no longer requested.
3. Offloads collected data from the data space to VSAM files.
4. Enables collection of data for new options.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2147I Press Enter to confirm REFRESH ALL or PF12 to cancel

Explanation: Confirmation that the Collector is to use the updated CICS IA options is required.

System action: If Enter is pressed, the Collector sends a REFRESH command to all active regions.

User response: Press Enter or PF12.

Module: CIUA100C

Destination

Terminal end user.

CIU2148I CINT *applid* CICS Resource options are updated

Explanation: The CINT transaction has successfully amended the CICS Resource operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA240C

Destination

Terminal end user and CINT TD queue.

CIU2149I CINT *applid* DB2/MQ/IMS options are updated

Explanation: The CINT transaction has successfully amended the DB2/MQ/IMS operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA250C

Destination

Terminal end user and CINT TD queue.

CIU2150I CINT *applid* General options are updated

Explanation: The CINT transaction has successfully amended the General operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA260C

Destination

Terminal end user and CINT TD queue.

CIU2151I CINT *applid* Time and Date options are updated

Explanation: The CINT transaction has successfully amended the Time and Date operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA280C

Destination

Terminal end user and CINT TD queue.

CIU2152I CINT *applid* CICS Affinity options are updated

Explanation: The CINT transaction has successfully amended the CICS Affinity operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA270C

Destination

Terminal end user and CINT TD queue.

CIU2153I CINT *applid* Global options are updated

Explanation: The CINT transaction has successfully amended the Global operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA300C

Destination

Terminal end user and CINT TD queue.

CIU2154I CINT *applid* Task options are updated

Explanation: The CINT transaction has successfully amended the Task operation options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA295C

Destination

Terminal end user and CINT TD queue.

CIU2155I CINT *applid* Natural resource options are updated

Explanation: The CINT transaction has successfully updated the Natural resource options.

System action: The operation options held on VSAM control file CIUCNTL are updated.

User response: None.

Module: CIUA29NC

Destination

Terminal end user and CINT TD queue.

| **CIU2157W Include list name is invalid**

| **Explanation:** The name of the TRUEs include list that had been input on the screen is invalid.

| **System action:** The CINT options will not be changed unless all the input is correct.

| **User response:** Correct the invalid input.

| **Module:** CIUA260C

| **Destination**

| Terminal end user.

| **CIU2159W Invalid key was pressed**

| **Explanation:** A function key pressed in response to a screen displayed by the CINC transaction is not valid for this screen.

| **System action:** The function key is ignored.

| **User response:** Use the correct function key.

| **Module:** CIUACM10

| **Destination**

| Terminal end user.

| **CIU2160I CINC session has ended**

| **Explanation:** The CINC transaction has ended.

| **System action:** The state of the Collector is unchanged.

| **User response:** None.

| **Module:** CIUACM10

| **Destination**

| Terminal end user and CINT TD queue.

| **CIU2161I CICS is terminating**

| **Explanation:** The CINC transaction has detected that CICS is terminating.

| **System action:** CICS IA is stopped.

| **User response:** None.

| **Module:** CIUACM10

| **Destination**

| Terminal end user and CINT TD queue.

| **CIU2162W Control file CIUCNTL is not available**

| **Explanation:** The CICS IA Control file, CIUCNTL, is disabled.

| **System action:** None.

| **User response:** Contact your CICS system support person.

| **Module:** CIUACM10

| **Destination**

| Terminal end user.

| **CIU2163W Error opening Control file CIUCNTL**

| **Explanation:** CICS IA encountered an error while trying to open the CIUCNTL control file.

| **System action:** None.

| **User response:** Contact your CICS system support person.

| **Module:** CIUACM10

| **Destination**

| CINT TD queue and terminal end user.

| **CIU2164W There are no connected CICS regions**

| **Explanation:** **Prompt for regions** request did not find any connected regions for your local CICS region.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM20

| **Destination**

| Terminal end user.

| **CIU2165W Too large number of selected regions**

| **Explanation:** More than one region was selected.

| **System action:** The input is rejected.

| **User response:** Select only one region.

| **Module:** CIUACM40

| **Destination**

| Terminal end user.

| **CIU2166W Invalid selection of APPLID**

| **Explanation:** The field with no APPLID was selected.

| **System action:** The input is rejected.

| **User response:** Select the field that contains an APPLID.

| **Module:** CIUACM40

| **Destination**

| Terminal end user.

| **CIU2168W Option option is invalid**

| **Explanation:** The *Option* input field on a screen is invalid because it contains an embedded blank space character, or starts with a digit, or has more than one wildcard character.

| **System action:** The input is rejected.

| **User response:** Correct the invalid input.

| **Module:** CIUACM10, CIUACM20

| **Destination**

| Terminal end user.

CIU2170W Options must be 1, 2, 3 or N(No)

Explanation: The valid values for this option are 1, 2, 3 or N(No).

System action: The value changes only if the input is correct.

User response: Correct the invalid input.

Module: CIUA300C

Destination

Terminal end user.

CIU2201S CICS command data failed RESP=eibresp RESP2=eibresp2 RCODE=eibrcode

| **Explanation:** Transaction CINT, CINB, or CINC received an invalid response when issuing EXEC CICS *command*. The response is in *eibresp*, *eibresp2* and *eibrcode*. Other *data*, if present, might give the object operated on by the *command*.

System action: The transaction is terminated by abending IUZA and the Collector is stopped.

User response: For further details of the exception *eibresp* refer to the *command* in the *CICS Application Programming Reference* manual or the *CICS System Programming Reference* manual.

For further information on how to determine system problems refer to the *CICS Problem Determination Guide*.

| **Module:** CIUA00HC, CIUA000C, CIUA100C,

CIUA110C, CIUA120C, CIUA130C, CIUA140C,
 CIUA150C, CIUA160C, CIUA200C, CIUA240C,
 | CIUA250C, CIUA260C, CIUA300C, CIUA400C,
 | CIUA410C, CIUA420C, CIUA440C, CIUA900C,
 | CIUACM00, CIUACM10, CIUACM20, CIUACM30,
 | CIUACM31, CIUACM32, CIUACM40, CIUACM60,
 | CIUACM61, CIUACM70, CIUACM71, CIUATM03,
 | CIUCINBE, CIUCINB1, CIUCINCE

Destination

CINT TD queue.

CIU2202S VSAM filetype file filename command
 failed RESP=eibresp RESP2=eibresp2

| **Explanation:** Transaction CINT, CINB or CINC received an invalid response when issuing EXEC CICS command on VSAM filetype file filename. The response is in eibresp and eibresp2.

System action: The transaction is terminated by abending IUZB and the Collector is stopped.

User response: For further details of the exception eibresp refer to the command in the CICS Application Programming Reference manual or the CICS System Programming Reference manual.

For further information on how to determine system problems refer to the CICS Problem Determination Guide.

Module: CIUA000C, CIUA110C, CIUA120C,
 CIUA130C, CIUA140C, CIUA200C, CIUA240C,
 | CIUA250C, CIUA260C, CIUA300C, CIUA400C,
 | CIUCINB2, CIUACM10, CIUACM20, CIUACM31,
 | CIUACM60, CIUACM61, CIUACM70, CIUACM71,
 | CIUATM01, CIUATM03

Destination

CINT TD queue.

CIU2204I CINT being used by userid

Explanation: User userid is using the CINT transaction.

System action: This user has exclusive use of the CINT transaction.

User response: None.

Module: CIUA000C

Destination

CINT TD queue.

CIU2206S CICS command PROGRAM program
 failed RESP=eibresp RCODE=eibrpcode

| **Explanation:** Transaction CINT, CINB or CINC received an invalid response when issuing command EXEC CICS command for Collector user exit program program.

System action: The transaction is terminated by abending IUZF and the Collector is stopped.

User response: For further details of the exception eibresp refer to the command in the CICS System Programming Reference manual.

For further information on how to determine system problems refer to the CICS Problem Determination Guide.

| **Module:** CIUA110C, CIUA120C, CIUA130C,
 CIUA140C, CIUA200C, CIUCINB1, CIUACM61

Destination

CINT TD queue.

CIU2207E DB2 table error on tablename SQL code code

Explanation: SQL error code code has occurred while querying DB2 table tablename.

System action: Program terminates normally but some expected output might be missing.

User response: Ensure that CICS IA is properly installed. Contact your system support group.

Module: CIUCINB2

Destination

CINT TD queue.

CIU2208E CICS region is not connected to DB2

Explanation: CICS has detected that there is no DB2 connection.

System action: Program terminates normally but some expected output might be missing.

User response: Contact your CICS system support person.

Module: CIUCINB2

Destination

CINT TD queue.

CIU2209S Records in control file CIUCNTL have incorrect format

| **Explanation:** Transaction CINT or CINC did not recognize the records in the control file.

System action: The transaction is terminated by abending IUZ4.

User response: If the control file was created by an earlier release of CICS IA, run any required migration job. If this does not solve the problem, delete and recreate the control file.

| **Module:** CIUA000C, CIUACM10

Destination

CINT TD queue.

CIU2210S Create dataspace action failed
REASON=reason code ERROR=error code

Explanation: The CINT transaction received an invalid response when issuing a call to the table manager, CIUTABM to create the MVS data space when starting the Collector.

System action: The transaction is terminated by abending IUZH and the Collector is stopped.

User response: The *reason code* value can be looked up in "Collector table manager diagnostics" on page 314. If it is AUTM_DSPSERV_CREATE_ERROR then *error code* is the value of GPR 0 after the MVS DSPSERV CREATE call. If it is AUTM_ALESERV_ADD_ERROR then *error code* is the value of GPR 15 after the MVS ALESERV ADD call. Use the appropriate MVS manual to find out the meaning of the error code.

Module: CIUA110C

Destination

CINT TD queue.

CIU2211S Create table action failed
REASON=reason code TABLE=table number

Explanation: The CINT transaction received an invalid response when issuing a create table call to the table manager, CIUTABM, for table *table number*.

System action: The transaction is terminated by abending IUZI and the Collector is stopped.

User response: The *reason code* and *table number* values can be looked up in "Collector table manager diagnostics" on page 314.

Module: CIUA110C, CIUA140C

Destination

CINT TD queue.

CIU2212S Add element action failed
REASON=reason code TABLE=table number

Explanation: The transaction CINT received an invalid response when issuing an add element call to the table manager, CIUTABM, for table *table number*.

System action: The transaction is terminated by abending IUZJ and the Collector is stopped.

User response: The *reason code* and *table number* values can be looked up in "Collector table manager diagnostics" on page 314.

Module: CIUA110C

Destination

CINT TD queue.

CIU2213S Stop of collector has failed during Start processing

Explanation: The Collector had failed and an attempt to restart it has failed again.

System action: The state of the Collector is unchanged.

User response: Contact your systems programmer.

Module: CIUA000C

Destination

CINT TD queue and terminal end user..

CIU2214I Collector is now state

Explanation: The CINT transaction has changed the Collector state to *state*.

System action: The state of the Collector is now *state*.

User response: None.

Module: CIUA000C, CIUA110C, CIUA120C, CIUA130C, CIUA140C

Destination

CINT TD queue.

CIU2215S Stop of Collector has failed

Explanation: An error has occurred whilst issuing an EXEC CICS DISABLE EXITALL for one of the exit programs.

System action: Review CINT log for message CIUX2239S to find the exit program. Retry the 'STOP' process or perform the 'START' process.

User response: Retry the 'STOP' process or do a 'START' of CICS IA if required.

Module: CIUA120C

Destination

CINT TD queue.

CIU2216S Destroy pool action failed
REASON=reason code ERROR=error code

Explanation: The transaction CINT received an invalid response when issuing a call to the table manager, CIUTABM, to destroy the MVS data space when stopping the Collector.

System action: The transaction is terminated by abending IUZN and the Collector is stopped.

User response: The *reason code* value can be looked up in "Collector table manager diagnostics" on page 314. If

it is AUTM_DSPSERV_DELETE_ERROR then *error code* is the value of GPR 0 after the MVS DSPSERV DELETE call. If it is AUTM_ALESERV_DELETE_ERROR then *error code* is the value of GPR 15 after the MVS ALESERV DELETE call. Use the appropriate MVS manual to find out the meaning of the error code.

Module: CIUA140C

Destination

CINT TD queue.

CIU2217S Destroy table action failed
REASON=reason code TABLE=table number

Explanation: The transaction CINT received an invalid response when issuing a destroy table call to the table manager, CIUTABM, for table *table number*.

System action: The transaction is terminated by abending IUZO and the Collector is stopped.

User response: The *reason code* and *table number* values can be looked up in “Collector table manager diagnostics” on page 314.

Module: CIUA140C

Destination

CINT TD queue.

CIU2218S CINT has abended *abend code* **in program** *program*

Explanation: This message is issued when the CINT HANDLE ABEND exit program is driven to handle a transaction abend that occurred within the CINT transaction. The abend code is given by *abend code*, and the failing program is given by *program*. Note that abends are issued by CINT, codes IUxx, as well as by CICS.

System action: The transaction is terminated with a transaction dump, with a dumpcode of *abend code*, and the Collector is stopped.

User response: If the original abend was issued by CINT, then there will be a preceding message on the CINT TD queue describing the abend. If so, refer to the description for that message. Otherwise, the abend was issued by CICS, for example ASRA, so refer to the *CICS Messages and Codes*.

Module: CIUCINTE

Destination

CINT TD queue and console.

CIU2219I Data type collector is now state

Explanation: The CINT transaction has changed the state of the *Data typeCollector* to *state*, where *Data type* is either “Dependency” or “Affinity”.

System action: The state of the *Data typeCollector* is now *state*.

User response: None.

Module: CIUA000C, CIUA110C, CIUA120C, CIUA130C, CIUA140C

Destination

CINT TD queue.

CIU2220S Create CPOOL action failed
REASON=reason code

Explanation: The CINT transaction received an invalid response when issuing a call to the CINB request queue manager, CIUCINP, to create its storage in the MVS CPOOL.

System action: The transaction is terminated by abending IUZQ and the Collector is stopped.

User response: The *reason code* value can be looked up in “Collector CINB request queue manager diagnostics” on page 316. Check that there was sufficient storage in your system for at least 4 KB of MVS CPOOL.

Module: CIUA110C

Destination

CINT TD queue.

CIU2221S Function call failed:
FUNCTION=function code
REASON=reason code

Explanation: Transaction CINT or CINB received an invalid response when issuing a call to the CINB request queue manager, CIUCINP, to perform function *function code*.

System action: The transaction is terminated by abending IUZR and the Collector is stopped.

User response: The *reason code* and *function code* values can be looked up in “Collector CINB request queue manager diagnostics” on page 316.

Module: CIUA120C, CIUA130C, CIUCINB1

Destination

CINT TD queue.

CIU2222S Destroy CPOOL action failed
REASON=reason code

Explanation: The CINT transaction received an invalid response when issuing a call to the CINB request queue manager, CIUCINP, to destroy its MVS CPOOL storage.

System action: The transaction is terminated by abending IUZS and the Collector is stopped.

User response: The *reason code* value can be looked up in “Collector CINB request queue manager diagnostics” on page 316.

Module: CIUA120C

Destination

CINT TD queue.

CIU2224S Error calculating space utilisation

Explanation: An error occurred during the calculation of the percentage of the data space currently occupied by dependency data.

System action: The transaction is terminated by abending IUZU and the Collector is stopped.

User response: Contact IBM support.

Module: CIUA150C

Destination

CINT TD queue.

CIU2225E Unsupported type of CINT task initiation

Explanation: The CINT transaction has been initiated in a way which is not allowed. The only valid ways to initiate a CINT transaction are:

- From a terminal
- From a console
- By issuing EXEC CICS START TRANSID('CINT') from another task.

System action: The transaction is terminated by abending IUZV and the Collector is stopped.

User response: Use one of the methods above to initiate CINT.

Module: CIUA000C

Destination

CINT TD queue

CIU2226E Incorrect CINT action: *action*

Explanation: The CINT transaction received an incorrect value for <action> when it was started by another task using EXEC CICS START TRANSID('CINT') FROM(<action>), or started from a terminal by entering CINT <action>. Only these <action> values are acceptable:

- START
- STOP
- PAUSE
- CONTINUE

- REFRESHOPTIONS
- STARTALL
- STOPALL
- PAUSEALL
- CONTINUEALL
- REFRESHALLOPTIONS
- STARTAFF
- STARTINT
- STARTBOTH
- STARTALLAFF
- STARTALLINT
- STARTALLBOTH

System action: The CINT transaction is stopped.

User response: Correct the <action> passed to CINT to be one of those above.

Module: CIUA000C

Destination

CINT TD queue and terminal end user, if started from a terminal.

CIU2227I Non-terminal CINT task initiating

Explanation: Transaction CINT has been initiated as a background non-terminal task.

System action: CINT runs in the background.

User response: None.

Module: CIUA000C

Destination

CINT TD queue

**CIU2228S Replace element action failed
REASON=*reason code* TABLE=*table number***

Explanation: Transaction CINT or CINB received an invalid response when issuing a call to the table manager, CIUTABM, to replace a table element for table *table number*.

System action: The transaction is terminated by abending IUZY and the Collector is stopped.

User response: The *reason code* and *table number* values can be looked up in “Collector table manager diagnostics” on page 314.

Module: CIUCINB2

Destination

CINT TD queue

CIU2230S VSAM dependency file *filename* header
 READ failed RESP=*eibresp*
 RESP2=*eibresp2*

Explanation: Transaction CINT received an invalid response when issuing an EXEC CICS READ command for the header record on VSAM dependency data file *filename*, when the Collector was starting with the restore data option set to Y. Either an incorrect file has been allocated for the dependency data file *filename* or the file is empty.

System action: The transaction is terminated by abending IUZ1 and the Collector is stopped.

User response: Check that the correct file is allocated, and that the Collector has previously been started with the same files. The first time the Collector is started, the CINT restore data option must be N. If the restore data option is initially set to Y, then CINT will abend with this message, because initially the VSAM dependency data file will be completely empty. The dependency file header records are added by CINT after the Collector has been started for the first time. For further details of the exceptions *eibresp* and *eibresp2* refer to the READ command in the *CICS Application Programming Reference* manual. For further information on how to determine system problems refer to the *CICS Problem Determination Guide*.

Module: CIUA110C

Destination

CINT TD queue

CIU2231I Number of records restored = *count*

Explanation: The Collector has been started with the restore option set to Y. The message gives the number of dependency records that were restored from the VSAM dependency data file to the MVS data space.

System action: The Collector is now RUNNING.

User response: None.

Module: CIUA110C

Destination

CINT TD queue.

CIU2232E CICS Release *release* is not supported by the Collector

Explanation: Transaction CINT or CINB has been initiated on a version/release/modification of CICS which the Collector does not support.

System action: The transaction is terminated by abending IUZ3 and the Collector is stopped.

User response: The Collector cannot be run on this CICS release.

Module: CIUA110C, CIUCINB1

Destination

CINT TD queue

CIU2233W CIUINTDT call failed REASON=*reason code*

Explanation: The CIU date formatter program, CIUINTDT, was unable to format the packed Julian date passed to it by its caller.

System action: Question marks are used for the date instead.

User response: The *reason code* value can be looked up in "Date formatter diagnostics" on page 316.

Module: CIUA150C, CIUREPPM

Destination

CINT TD queue

CIU2236I EXEC CICS DISABLE STOP for program *exitprog* OK

Explanation: The EXEC CICS DISABLE STOP command for program *exitprog* was successful.

System action: None.

User response: None.

Module: CIUA120C, CIUA130C

Destination

CINT TD queue.

CIU2238I EXEC CICS DISABLE EXITALL for program *exitprog* OK

Explanation: The EXEC CICS DISABLE EXITALL command for program *exitprog* was successful.

System action: None.

User response: None.

Module: CIUA120C

Destination

CINT TD queue.

CIU2239S DISABLE EXITALL for program *exitprog* failed RESP=*eibresp* RCODE=*reason code*

Explanation: The EXEC CICS DISABLE EXITALL command for program *exitprog* was unsuccessful.

System action: Processing of 'STOP' of CICS IA continues until all exits are stopped. The 'STOP' process is then flagged as 'STOP FAILED' and message CIU2215S is issued.

User response: Retry the 'STOP' process or do a 'START' of CICS IA if required.

Module: CIUA120C

Destination

CINT TD queue.

CIU2240I EXEC CICS ENABLE EXIT for program *exitprog*, EXIT *exit* OK

Explanation: The EXEC CICS ENABLE EXIT command for program *exitprog* at exit point *exit* was successful.

System action: None.

User response: None.

| Module: CIUA110C, CIUACM61

Destination

CINT TD queue.

CIU2241I EXEC CICS ENABLE EXIT START for program *exitprog* OK

Explanation: The EXEC CICS ENABLE EXIT START command for program *exitprog* was successful.

System action: None.

User response: None.

| Module: CIUA110C, CIUACM61.

Destination

CINT TD queue.

CIU2242I EXIT PROGRAM *exitprog* already disabled

Explanation: The exit program *exitprog* is already disabled or has not been enabled.

System action: None.

User response: None.

Module: CIUA110C

Destination

CINT TD queue.

CIU2243I EXEC CICS ENABLE TRUE for program *exitprog* OK

Explanation: The EXEC CICS ENABLE EXIT command for the task-related user exit program *exitprog* was successful.

System action: None.

User response: None.

| Module: CIUA110C, CIUACM61.

Destination

CINT TD queue.

CIU2244S Exclude list *name* has invalid contents at offset *offset*

Explanation: When transaction CINT attempted to start the Collector it found that the required program or transaction exclude list had invalid contents at the offset shown.

System action: The transaction is terminated by abending IUZ5.

User response: Correct the error in the exclude list and try again.

Module: CIUA110C

Destination

CINT TD queue.

CIU2245S Dump domain function *function* failed RESP=*resp* RCODE=*rancode*

Explanation: Unexpected error detected by CICS dump domain function.

System action: The transaction is terminated by abending IUZ6.

User response: Contact IBM support.

| Module: CIUA110C, CIUA120C, CIUACM61,
| CIUACM71

Destination

CINT TD queue

CIU2246E Collector is not in START or PAUSED state

Explanation: The Collector must be in the START or PAUSED state for the selected option.

System action: The state of the Collector is not changed.

User response: Make sure that the Collector is started and try the option again.

Module: CIUA000C, CIUA110C

Destination

Terminal end user.

| **CIU2247W CICS IA dataspace is *nn* percent full**

| **Explanation:** The MVS data space preallocated for the
| Collector is *nn* percent full. The message is issued if
| over 80% of the MVS data space is filled.

| **System action:** None.

| **User response:** None.

| **Module:** CIUCINB1

Destination

CINT TD queue and console.

CIU2248I CINT *applid* CICS Resource options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed CICS Resource options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA240C

Destination

CINT log.

CIU2249I CINT *applid* DB2/MQ/IMS options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed DB2, MQ, or IMS options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA250C

Destination

CINT log.

CIU2250I CINT *applid* General options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed General options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA260C

Destination

CINT log.

CIU2251I CINT *applid* Time and Date options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed Time and Date options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA280C

Destination

CINT log.

CIU2252I CINT *applid* CICS Affinity options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed CICS Affinity options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA270C

Destination

CINT log.

CIU2253I CINT *applid* Global options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed Global options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA300C

Destination

CINT log.

CIU2254I CINT *applid* Task options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with changed Task options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUA295C

Destination

CINT log.

CIU2255I CINT *applid* Natural resource options are updated by *userid*

Explanation: The control record for the supplied APPLID is updated with the changed NATURAL resource options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

CIU2256I • CIU2278S

Module: CIUA29NC

Destination

CINT log.

CIU2256I CINT *command_name* is requested by *userid* for *applid*

Explanation: A *command* for the supplied APPLID is requested by user *userid*.

System action: None.

User response: None.

Module: CIUA000CC, CIUA100C

Destination

CINT TD queue.

CIU2257I CINT Collector runtime options for *applid*:

Explanation: Collector runtime and Global options for the supplied APPLID.

System action: None.

User response: None.

Module: CIUA105CC, CIUA110C

Destination

CINT TD queue.

CIU2258I *options_list*

Explanation: This message contains the list of the following collection options for the supplied region:

- The Collector runtime options, when it is displayed together with the message CIU2257I.
- The Collector changed options, when it is displayed together with one of the messages CIU2248I through CIU2255I.

System action: None.

User response: None.

Module: CIUA105CC, CIUA110C, CIUA240C, CIUA250C, CIUA260C, CIUA270C, CIUA295C, CIUA29NC, CIUA300C

Destination

CINT TD queue.

| **CIU2275W** Control file CIUCNTL not open in the FOR

| **Explanation:** CICS IA encountered an error while trying to open the CIUCNTL control file in the file-owning region.

| **System action:** The CINC transaction is stopped.

| **User response:** Contact your CICS system support.

| **Module:** CIUACM10

Destination

| CINT TD queue and terminal end user.

| **CIU2276S** CICS *command data* failed RESP=*eibresp* RESP2=*eibresp2* RCODE=*eibrcode*

| **Explanation:** The CINC transaction received an invalid response when issuing EXEC CICS *command*. The response is in *eibresp*, *eibresp2* and *eibrcode*. Other *data*, if present, might give the object operated on by the *command*.

| **System action:** The CINC transaction continues. A message is sent to the terminal. The requested action fails.

| **User response:** For further details of the exception *eibresp* refer to the *command* in the CICS Application Programming Reference manual or the CICS System Programming Reference manual. For further information on how to determine system problems refer to the CICS Problem Determination Guide.

| **Module:** CIUACM10

Destination

| Terminal end user and CINT TD queue.

| **CIU2277S** USER record is not found in the CIUCNTL control file

| **Explanation:** A USER record was accidentally deleted.

| **System action:** The transaction is terminated with the IUXD abend code.

| **User response:** Create the USER record again. If the problem recurs, contact the IBM support.

| **Module:** CIUACM20, CIUACM31, CIUACM60, CIUACM70

Destination

| CINT TD queue.

| **CIU2278S** USER record in the CIUCNTL control file has incorrect format

| **Explanation:** A USER type record for a specified user in the CIUCNTL control file has incorrect format.

| **System action:** The transaction is terminated with the IUXE abend code.

| **User response:** Contact your CICS system support person.

| **Module:** CIUACM10

Destination

| CINT TD queue.

CIU2279S CONTROL1 record is not found in the CIUCNTL control file.

Explanation: A CONTROL1 type record is not found in the CIUCNTL control file.

System action: The transaction is terminated with the IUXY abend code. The command flow collector is stopped.

User response: Contact your CICS system support person.

Module: CIUACM10

Destination

CINT TD queue.

CIU2280S REGION record is not found in the CIUCNTL control file.

Explanation: A REGION type record is not found in the CIUCNTL control file.

System action: The transaction is terminated with the IUXV abend code.

User response: Contact your CICS system support person.

Module: CIUACM10

Destination

CINT TD queue.

CIU2281S CICS command data failed RESP=eibresp RESP2=eibresp2 RCODE=eibrcode

Explanation: This message is issued when the CINC HANDLE ABEND exit program is driven to handle a transaction abend that occurred within the CINC transaction. The abend code is given by *abend code*, and the failing program is given by *program*. Note that abends are issued by CINC, codes IUxx, as well as by CICS.

System action: The transaction is terminated with a transaction dump, with a dumpcode of *abend code*, and the Collector is stopped.

User response: If the original abend was issued by CINC, there will be a preceding message on the CINT TD queue describing the abend. If so, refer to the description for that message. Otherwise, the abend was issued by CICS, for example ASRA, so refer to the *CICS Messages and Codes*.

Module: CIUA171C, CIUCINCE

Destination

CINT TD queue and terminal end user.

CIU2283I CINC being used by userid

Explanation: User *userid* is using the CINC transaction.

System action: None.

User response: None.

Module: CIUACM10

Destination

CINT TD queue.

CIU2284S Records in control file CIUCNTL have incorrect format

Explanation: The CINC transaction did not recognize the records in the control file.

System action: The transaction is not terminated. The abend code is IUZ4.

User response: If the Control file was created by an earlier release of CICS IA, run any required migration job. If this does not solve the problem, delete and create the Control file again.

Module: CIUACM10

Destination

CINT TD queue.

CIU2285E Unsupported type of CINC task initiation

Explanation: The CINC transaction has been initiated in a way, which is not allowed. The CINC transaction can be initiated only from the 3270 terminal.

System action: The transaction is terminated by abending IUXF.

User response: Use the proper methods to initiate CINC.

Module: CIUACM10

Destination

CINT TD queue.

CIU2287E CICS Release *release* is not supported by the CINC

Explanation: Transaction CINC has been initiated on a version, release or modification of CICS, which the CINC Collector does not support.

System action: The transaction is terminated with the IUXG abend code.

User response: The CINC Collector cannot be run on this CICS release.

Module: CIUACM10, CIUACM32, CIUACM61, CIUACM71

Destination

CINT TD queue.

CIU2288S CINC abending - internal error elsewhere in the CINC

Explanation: The internal error encountered elsewhere, either in CINC or a Command Flow collector exit program.

System action: The transaction is terminated with the IUXX abend code. The Command Flow collector is stopped.

User response: Refer to any earlier messages on the CINC TD queue for the cause of the error.

Module: CIUACM10

Destination

CINT TD queue.

CIU2289I CINC command flow options are updated by userid

Explanation: The control record for the supplied USERID is updated with changed Command Flow options. This record is stored in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUACM10, CIUACM20

Destination

Terminal end user and CINT TD queue.

CIU2290I options_list

Explanation: This message contains the list of the following collection options for the supplied USERID:

- The Collector changed options, when it is displayed together with the CIU2289I message.
- The Collector run time options, when it is displayed together with the CIU2296I message.

System action: None.

User response: None.

Module: CIUACM10, CIUACM20, CIUACM61

Destination

CINT TD queue.

CIU2291S CICS command PROGRAM program failed RESP=eibresp RCODE=eibrcode

Explanation: The CINC transaction received an invalid response when issuing command EXEC CICS *command* for CINC Collector user exit program *program*.

System action: The transaction is terminated with the IUXT abend code. The Collector is not stopped.

User response: For further details of the exception *eibresp* refer to the *command* in the *CICS System Programming Reference* manual. For further information on how to determine system problems refer to the *CICS Problem Determination Guide*.

Module: CIUACM61

Destination

CINT TD queue.

CIU2292I EXEC CICS ENABLE EXIT for program exitprog, EXIT exit OK

Explanation: The EXEC CICS ENABLE EXIT command for the *exitprog* program at the *exit* exit point was successful.

System action: None.

User response: None.

Module: CIUACM61

Destination

CINT TD queue.

CIU2293W CINC Collector is already active for userid

Explanation: The CINC Collector is already active for a user when an attempt to start another collection session for the same user on the same CICS region is made.

System action: The second Collector start request is terminated because only one instance of collection session at a time is permitted for each user on the same CICS region.

User response: Find out on which region the CINC Collector is currently in use. Only one Collector session can be started by each user on each CICS region at a given time.

Module: CIUACM60, CIUACM61

Destination

CINT TD queue.

CIU2294W Max concurrent CINC Collector sessions limit reached

Explanation: The CINC Collector session was started and received a response from the start request service, CIUACM61, that it was unable to receive the start request from a user, because the maximum limit of concurrent CINC Collector sessions was reached.

System action: The CINC Collector start request is rejected.

| **User response:** Decrease the number of concurrent CINC Collector sessions on a region.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2295W** **User journal name *journal_name* value is already used by other active user**

| **Explanation:** The CINC Collector session was started and received a response from the start request service, CIUACM61, that it was unable to receive the start request from a user, because the user journal name value is already used by other active CINC users.

| **System action:** The CINC Collector start request is rejected.

| **User response:** Change the value for a user journal name parameter.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2296I** **CINC Collector runtime options for *userid***

| **Explanation:** The CINC Collector runtime options for the supplied USERID.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2297W** **CINC Collector is not started in *all/some* your regions**

| **Explanation:** An attempt to start the Collector from CINC was made. However, the Collector was not currently in an appropriate state to make the change. See details in the CINT TD queue of CICS regions.

| **System action:** The Collector state is not changed.

| **User response:** Check why the Collector is currently in an improper state.

| **Module:** CIUACM60

| **Destination**

| Terminal end user.

| **CIU2298I** **CINC Collector is started in your regions**

| **Explanation:** An attempt to start the Collector from CINC was made.

| **System action:** The Collector state is changed to running in all the regions from your configuration.

| **User response:** None.

| **Module:** CIUACM60

| **Destination**

| Terminal end user.

| **CIU2299W** **CINC Collector is not started. All options are empty.**

| **Explanation:** An attempt to start the Collector from CINC was made. However, all the Collector options are empty.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM60, CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2300W** **CINC Collector is not started. *Option* option is empty**

| **Explanation:** An attempt to start the Collector from CINC was made. However, the *option* option is empty.

| **System action:** The CINC Collector start request is rejected.

| **User response:** Change values for the CINC Collector option.

| **Module:** CIUACM60

| **Destination**

| Terminal end user.

| **CIU2301W** **CINC Collector is not started. Some regions are not connected.**

| **Explanation:** An attempt to start the Collector from CINC was made. However, all or some of the regions from your configuration are not connected to the local CICS region.

| **System action:** The CINC Collector start request is rejected.

| **User response:** Check why the region configuration is in an inappropriate state.

| **Module:** CIUACM60

| **Destination**

| Terminal end user.

| **CIU2302I CINC Collector is stopped successfully.**

| **Explanation:** An attempt to stop the Collector from CINC was made.

| **System action:** The Collector state is changed to stopped in all the regions from your configuration.

| **User response:** None.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2303W CINC Collector is not stopped. There are no active collector sessions.**

| **Explanation:** An attempt to stop the Collector from CINC was made. However, there are no active collector sessions in the regions from your regions configuration.

| **System action:** The CINC Collector stop request is rejected.

| **User response:** None.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2304W CINC Collector is not stopped. There are no connected regions.**

| **Explanation:** An attempt to stop the Collector from CINC was made. However, all the regions from your configuration are not connected to or not defined for the local CICS region.

| **System action:** The CINC Collector stop request is rejected.

| **User response:** None.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2305W CINC Collector is not stopped. There are no regions in configuration.**

| **Explanation:** An attempt to stop the Collector from CINC was made. However, the list of regions in your configuration is empty.

| **System action:** The CINC Collector stop request is rejected.

| **User response:** None.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2306I Statistics request has ended.**

| **Explanation:** An attempt to start the CINC Collector statistics request was made.

| **System action:** The CINC Collector statistics data is formed for all the regions from your configuration.

| **User response:** None.

| **Module:** CIUACM31

| **Destination**

| Terminal end user.

| **CIU2307W Statistics request is not executed. There are no regions in configuration.**

| **Explanation:** An attempt to start the CINC Collector statistics request was made. However, the list of regions in your configuration is empty.

| **System action:** The Statistics request is rejected.

| **User response:** None.

| **Module:** CIUACM31

| **Destination**

| Terminal end user.

| **CIU2308W Statistics request failed.**

| **Explanation:** An attempt to start the CINC Collector statistics request was made. However, the successful termination of request is not possible.

| **System action:** The Statistics request is rejected.

| **User response:** Refer to any earlier messages on the CINC TD queue for the cause of the error.

| **Module:** CIUACM31

| **Destination**

| Terminal end user.

| **CIU2309I CINC Collector is stopped for *userid*:
Total Command flow records *number***

| **Explanation:** The CINC Collector is stopped for the supplied USERID.

| **System action:** The CINC Collector is stopped for the supplied USERID.

| **User response:** None.

| **Module:** CIUACM71

| **Destination**

| CINT TD queue.

| **CIU2310W CINC Collector is not stopped. Option option is empty.**

| **Explanation:** An attempt to Stop the Collector from CINC was made. However, the *option* option is empty.

| **System action:** The CINC Collector stop request is rejected.

| **User response:** Change values for CINC Collector option.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2311W CINC Collector is not started. Journal name name was not found.**

| **Explanation:** An attempt to start the Collector from CINC was made. However, the Journal name was not found in the region resource definitions.

| **System action:** The CINC Collector start request is rejected.

| **User response:** Change values for CINC Collector options.

| **Module:** CIUACM61

| **Destination**

| Terminal end user.

| **CIU2313W Invalid cursor position**

| **Explanation:** The cursor was set to the wrong position and the F4 function key was pressed.

| **System action:** None.

| **User response:** Place the cursor in the APPLID field and press F4.

| **Module:** CIUACM20

| **Destination**

| Terminal end user.

| **CIU2314W CINC already in use by user user**

| **Explanation:** The CINC transaction is already used by user *userid* when an attempt to start another CINC transaction by the same user is made.

| **System action:** The second CINC transaction is terminated, because only one instance of CINC is permitted for the same user.

| **User response:** Check where CINC is currently in use.

| Only one instance of CINC is permitted for the same user at a given time.

| **Module:** CIUACM10, CIUATM03

| **Destination**

| Terminal end user and CINT TD queue.

| **CIU2316W CINC Collector is not started. All regions are not connected**

| **Explanation:** The Collector is not started in all requested regions.

| **System action:** The request to start the Collector fails for all requested regions.

| **User response:** Ensure that all APPLIDs are correct and the corresponding regions are started and connected.

| **Module:** CIUACM60

| **Destination**

| Terminal end user.

| **CIU2317W CINC Collector is already active for some regions**

| **Explanation:** The Collector is already active for some regions from the APPLID list.

| **System action:** The request to start the Collector fails for the regions where the Collector is stopped.

| **User response:** Press F7 to get the statistics about the Collector state on the regions. Exclude the regions with the active Collector status from the APPLID list. Start the Collector again.

| **Module:** CIUACM60

| **Destination**

| Terminal end user and CINT TD queue.

| **CIU2319W Updates are rejected because CINC Collector is running for userid**

| **Explanation:** The updates are rejected because the Command Flow collector is running for the specified user.

| **System action:** The request to update the options or the APPLID list fails.

| **User response:** Stop the Collector. Update the options or the APPLID list, or both. Start the Collector.

| **Module:** CIUACM10, CIUACM20

| **Destination**

| Terminal end user.

| **CIU2320I** CINC Collector is stopped. Total records
| written: *number*

| **Explanation:** The Collector is stopped. The *number* is
| the total number of the Command Flow records written
| to a user journal during data collection.

| **System action:** The Collector is stopped. The
| Command Flow Statistics panel, CIUA03, appears. See
| Figure 32 on page 110.

| **User response:** None.

| **Module:** CIUACM70

| **Destination**

| Terminal end user.

| **CIU2322W** Duplicate APPLIDs specified. Updates
| are rejected.

| **Explanation:** The updates are rejected because
| duplicate APPLIDs were specified.

| **System action:** The updates are rejected.

| **User response:** Eliminate the duplicate APPLIDs.

| **Module:** CIUACM20

| **Destination**

| Terminal end user.

| **CIU2323W** Journal Copy Criteria option is invalid.
| Enter LAST, USER or CFID

| **Explanation:** An invalid value was entered for the
| Journal Copy Criteria option.

| **System action:** The updates are rejected.

| **User response:** Enter the correct value: LAST, USER or
| CFID.

| **Module:** CIUACM10

| **Destination**

| Terminal end user.

| **CIU2324W** CINC Collector is not started in all your
| regions

| **Explanation:** The Collector is not started on all your
| regions.

| **System action:** The start Collector request failed on all
| the requested regions.

| **User response:** Press F7 to get the list of all regions.
| To detect the reason of the problem, check CINT TDQ
| on all regions for the messages related to this start
| Collector request.

| **Module:** CIUACM31, CIUACM60

| **Destination**

| Terminal end user.

| **CIU2325W** CINC Collector is started in some your
| regions

| **Explanation:** The Collector is started on some your
| regions.

| **System action:** The start Collector request failed in
| some of the requested regions.

| **User response:** Press F7 to get the list of all regions.
| To detect the reason of the problem, check CINT TDQ
| on the regions with the stopped Collector status for the
| messages related to this start Collector request.

| **Module:** CIUACM31, CIUACM60

| **Destination**

| Terminal end user.

| **CIU2326W** CINC was not completed for *userid*

| **Explanation:** The previous transaction CINC for user
| *userid* was not completed successfully.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM10

| **Destination**

| Terminal end user.

| **CIU2327W** CICS ApplIDs option is empty

| **Explanation:** No applIDs are defined.

| **System action:** The request is rejected.

| **User response:** Define at least one applID.

| **Module:** CIUACM31

| **Destination**

| Terminal end user.

| **CIU2328W** CINC Collector was not completed for
| *userid*

| **Explanation:** The Collector stopping was not
| completed for user *userid*. For example, the Collector
| was stopped from another region.

| **System action:** None.

| **User response:** Stop the Collector on the region where
| the message was encountered.

| **Module:** CIUACM31

| **Destination**

| Terminal end user.

| **CIU2329S** **An abend occurred in program *program* for region *region***

| **Explanation:** An abend occurred in the remote *region* region.

| **System action:** The Collector is not started in the remote region. The CINC transaction in a local region is terminated by abending IUXI.

| **User response:** Check CINT TD queue in the remote region to fix the problem. Start the CINC transaction again.

| **Module:** CIUACM31, CIUACM60, CIUACM70

| **Destination**

| CINT TD queue.

| **CIU2330S** **CINC Collector for USERID *userid1* is already started by *userid2***

| **Explanation:** The CINC collector with the option USERID *userid1* is already started by user *userid2*.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2351E** **Invalid journal type in model**

| **Explanation:** The TYPE attribute in the JOURNALMODEL definition is not MVS.

| **System action:** The Collector start failed.

| **User response:** Set the TYPE attribute (MVS) in the JOURNALMODEL definition.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2352E** **Different journal stream names**

| **Explanation:** The Collector is not started because of the log stream names inconsistency.

| **System action:** The request to start the Collector failed.

| **User response:** Correct the log stream names according to the following:

- | 1. The journal name must have the same name as the corresponding journal model.
- | 2. The log stream name in a given journal model must be the same on all regions.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2353E** **Different Control file names**

| **Explanation:** The Collector is not started because of the dataset names inconsistency for the CIUCNTL control file.

| **System action:** The request to start the Collector failed.

| **User response:** Check the CIUCNTL control file definition on all regions. It must point to the same dataset if the control file is shared between the regions.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2354I** **User exit *exit* was loaded**

| **Explanation:** The specified user exit was loaded.

| **System action:** None.

| **User response:** None.

| **Module:** CIUACM61

| **Destination**

| CINT TD queue.

| **CIU2500I** **There is no action to process.**

| **Explanation:** The message appears when Enter is pressed without any action specified.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2501W** **No more users allowed. Maximum limit of users reached.**

| **Explanation:** The maximum limit of users is already reached when an attempt to add or copy a user is made.

| **System action:** None.

| **User response:** Delete unnecessary users to clear the space for a new user.

| **Module:** CIUA400C, CIUA410C, CIUA420C

| **Destination**

| 3270

| **CIU2502I** **No users were added.**

| **Explanation:** The Add User Menu panel is closed without adding a new user.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2503I** **User *userid* was not copied.**

| **Explanation:** No user was copied when returning to the User Administration Menu panel, CIU400, from the Copy User Menu panel, CIU420.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2504E** **Cannot manage user *userid*. It is in use by someone else.**

| **Explanation:** The program cannot access the user record in the control file. The record is blocked because it is used by another person or program.

| **System action:** None.

| **User response:** Try the action again later.

| **Module:** CIUA400C, CIUA440C, CIUATM03

| **Destination**

| 3270, CINT TD queue

| **CIU2505E** **User *userid* has "ACTIVE" status. Stop CINC collector and try again.**

| **Explanation:** The Command Flow collector is active for the specified user. You cannot delete a user when the Command Flow collector has an ACTIVE status for this user.

| **System action:** None.

| **User response:** Stop the Command Flow collector and try the action again.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2506I** **User *userid* was not updated.**

| **Explanation:** The action specified on the User Details Menu panel, CIU440, is canceled.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2507W** **Confirm the deletion of user *userid*.**

| **Explanation:** To prevent accidental deletion of a user, confirm whether you are sure that the specified user is to be deleted.

| **System action:** None.

| **User response:** Confirm or cancel the deletion of the user.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2508I** **The deletion of user *userid* was cancelled.**

| **Explanation:** The user is not deleted.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2509W** **Journal name must begin with a letter.**

| **Explanation:** The journal name might be entered improperly. Check the first character and change it with a valid symbol.

| **System action:** None.

| **User response:** Correct the invalid input.

| **Module:** CIUA410C, CIUA420C, CIUA440C

| **Destination**

| 3270

| **CIU2510E** **Cannot delete user *userid* – CINC session is active.**

| **Explanation:** You can delete a user only when a CINC session for this user is closed.

| **System action:** None.

| **User response:** Quit the CINC transaction.

| **Module:** CIUA400C

| **Destination**

| 3270

| **CIU2511W** **User name can contain national characters. It must be alphanumeric.**

| **Explanation:** The specified user name might contain national characters or other non-alphanumeric symbols. A user name must consist of alphanumeric characters.

| **System action:** None.

| **User response:** Type the user name using proper characters.

| **Module:** CIUA410C, CIUA420C, CIUATM03

| **Destination**

| 3270

| **CIU2512W** **Journal name can contain national characters. It must be alphanumeric.**

| **Explanation:** The specified journal name might contain national characters or other non-alphanumeric symbols. A user name must consist of alphanumeric characters.

| **System action:** None.

| **User response:** Type the journal name using proper characters.

| **Module:** CIUA410C, CIUA420C, CIUA440C

| **Destination**

| 3270

| **CIU2513E** **Cannot update user *userid* – CINC session is active.**

| **Explanation:** You can update user data only when the CINC transaction session for this user is closed.

| **System action:** None.

| **User response:** Quit the CINC transaction.

| **Module:** CIUA440C, CIUATM03

| **Destination**

| 3270

| **CIU2514I** **User *userid* was added.**

| **Explanation:** The user was successfully added.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA410C, CIUA420C, CIUATM03

| **Destination**

| 3270

| **CIU2515E** **Cannot add user *userid* – CINC user or CINT region already exists.**

| **Explanation:** A user or a region with the same name already exists.

| **System action:** None.

| **User response:** Enter another user name.

| **Module:** CIUA410C, CIUA420C, CIUATM03

| **Destination**

| 3270

| **CIU2516E** **Cannot add user *userid* – internal error occurred.**

| **Explanation:** An internal error is encountered when adding the user record to the control file.

| **System action:** None.

| **User response:** Try the action again.

| **Module:** CIUA410C, CIUA420C, CIUATM03

| **Destination**

| 3270 and CINT TD Queue

| **CIU2517W** **Cannot update user *userid* – CINC collector is active.**

| **Explanation:** You can update a user data only when the Command Flow collector has a NOT ACTIVE status for this user.

| **System action:** None.

| **User response:** Stop the Command Flow collector for this user.

| **Module:** CIUA440C, CIUATM03

| **Destination**

| 3270

| **CIU2518I** **User *userid* updated.**

| **Explanation:** The user data is successfully updated.

| **System action:** None.

| **User response:** None.

| **Module:** CIUA440C, CIUATM03

| **Destination**

| 3270

CIU2519E **Update failed – user *userid* does not exist.**

Explanation: The user record was deleted from the control file by another transaction during the administration of CINC users.

System action: None.

User response: Close panel CIU440 and refresh the users list.

Module: CIUA440C, CIUATM03

Destination

3270 and CINT TD Queue.

CIU2520E **Cannot update user *userid* – internal error occurred.**

Explanation: The user record data has a wrong format.

System action: None.

User response: Close panel CIU440 and refresh the users list.

Module: CIUA440C, CIUATM03

Destination

3270 and CINT TD Queue.

CIU2521E **CICS TS release is *current_release*. Release *required_release* or higher is required.**

Explanation: The used version of CICS TS is out of date.

System action: All the actions, except displaying user list, are blocked.

User response: Migrate to CICS TS version 3.1 or later.

Module: CIUA400C

Destination

3270 and CINT TD Queue.

CIU2522W **Consequences of deleting user with unknown status are unpredictable.**

Explanation: If the user status is unknown, you are asked to confirm or cancel the deletion of this user.

System action: None.

User response: Confirm or cancel the deletion.

Module: CIUA400C

Destination

3270 and CINT TD Queue.

CIU3117I **CINT options refreshed**

Explanation: This message is written in the CINT TDQ when the IA options are refreshed.

System action: None.

User response: None.

Module: CIUA105C

Destination

CINT TDQ.

CIU3301I **CINB task is starting**

Explanation: The CINB transaction has been initiated by CINT. CINB saves dependency data from the data space to the dependency data file.

System action: The Collector continues RUNNING.

User response: None.

Module: CIUCINB1

Destination

CINT TD queue

CIU3302S **CINB received an unrecognizable request**

Explanation: Transaction CINB received an invalid request to perform one of its functions from another component of the Collector, CINT or a Collector exit program.

System action: The transaction is terminated by abending IUYA and the Collector is stopped.

User response: Contact IBM support.

Module: CIUCINB1

Destination

CINT TD queue

CIU3303S **CINB has abended *abend code* in program *program***

Explanation: This message is issued when the CINB HANDLE ABEND exit program is driven to handle a transaction abend that occurred within the CINB transaction. The abend code is given by *abend code*, and the failing program is given by *program*. Note that abends are issued by CINB (codes IUxx) as well as by CICS.

System action: The transaction is terminated with a transaction dump, with a dumpcode of *abend code*, and the Collector is stopped.

User response: If the original abend was issued by CINB, then there will be a preceding message on the CINT TD queue describing the abend. If so, refer to the description for that message. Otherwise, the abend was

issued by CICS, for example, ASRA, so refer to the *CICS Messages and Codes*.

Module: CIUCINBE

Destination

CINT TD queue and console.

CIU3304S CINB abending - system error elsewhere in the Collector

Explanation: The transaction CINB received a request to abend from another component of the Collector. The request was sent because of an error encountered elsewhere, either in CINT or a Collector exit program.

System action: The transaction is terminated by abending IUYC and the Collector is stopped.

User response: Refer to any earlier messages on the CINT TD queue for the cause of the error.

Module: CIUCINB1

Destination

CINT TD queue

CIU3305I CINB save started - because of *reason*

Explanation: Transaction CINB is commencing a scan of the dependency table in the data space to write any data changed since the previous save to the dependency data file. The save might have been started for one of four possible *reasons*:

- Collector stopped (STOP)
- Save interval reached (TIME)
- Activity count reached (TRIGGER)
- Collector paused (PAUSE)

Note that the latter three reasons can only occur when the CINT perform periodic saves option is set to Y.

System action: CINB saves changed data elements from the data space to the dependency data file.

User response: None.

Module: CIUCINB2

Destination

CINT TD queue

CIU3306I CINB save ended - *count* records saved

Explanation: Transaction CINB has finished the scan of the dependency table in the data space and wrote *count* records to the dependency data file.

System action: The number of records given by *count* were saved to the dependency data file.

User response: If the CINT perform periodic saves option is set to Y, and *count* has been consistently near

zero for the past few saves, this might indicate that the Collector has detected all the dependencies it can and the Collector might be stopped.

Module: CIUCINB2

Destination

CINT TD queue

CIU3307I CINB terminated - Collector is stopping

Explanation: Transaction CINB received a request to terminate, from CINT, because the Collector is stopping.

System action: The transaction is terminated and the Collector is stopped.

User response: None.

Module: CIUCINB1

Destination

CINT TD queue

CIU3308I Message received from program *program*

Explanation: Transaction CINB received a message from program *program* to write to CINT TD queue.

System action: The associated message is written to CINT, which is the only mechanism available to the Collector exit programs when they wish to issue a message.

User response: Examine the following message on the CINT TD queue.

Module: CIUCINB1

Destination

CINT TD queue.

CIU3309I Collector *action* by date/time options

Explanation: The date/time options have caused the collector to be PAUSED or CONTINUED at this time.

System action: Collection is automatically PAUSED or CONTINUED as required by the date/time options.

User response: None.

Module: CIUCINB3

Destination

CINT TD queue.

CIU3310S Invalid file number for table in GWA

Explanation: Transaction CINB found an incorrect value, the dependency file number, in an internal array in the Collector GWA, suggesting that the GWA has been corrupted.

System action: The transaction is terminated by abending IUYE and the Collector is stopped.

User response: Attempt to find out the cause of the corruption. It could be due to an application accidentally overwriting the GWA.

Module: CIUA110C, CIUCINB2

Destination

CINT TD queue

CIU3311E Transaction CINB must be initiated by transaction CINT

Explanation: Transaction CINB could not have been initiated by CINT, as its CICS startcode indicates something other than EXEC CICS START with no data.

System action: The transaction is terminated by abending IUYF and the Collector is stopped.

User response: The CINB transaction can only be started by the Collector control transaction CINT.

Module: CIUCINB1

Destination

CINT TD queue

CIU3312S CINB abending - CICS is terminating

Explanation: Transaction CINB found that CICS had entered quiesce state before shutdown and the Collector was still RUNNING.

System action: The transaction is terminated by abending IUYG and the Collector is stopped.

User response: None. To avoid this, always stop the Collector before shutting down CICS. The CINT STOP action could be submitted from a CICS Shutdown PLT program, using EXEC CICS START.

Module: CIUCINB1

Destination

CINT TD queue

CIU3313S Invalid address for program in the GWA

Explanation: Transaction CINT or CINB found that the address of Collector program *program* in the GWA was a null value, suggesting that the GWA has been corrupted.

System action: The transaction is terminated by abending IUYH and the Collector is stopped.

User response: Attempt to find out the cause of the corruption. It could be due to an application accidentally overwriting the GWA.

Module: CIUA120C, CIUA130C, CIUCINB1

Destination

CINT TD queue

CIU3314S Function call failed FUNCTION=*function code* REASON=*reason code* TABLE=*table number*

Explanation: Transaction CINT or CINB received an invalid response when issuing a call to the table manager, CIUTABM, to access a table element for table *table number*.

System action: The transaction is terminated by abending IUYI and the Collector is stopped.

User response: The *reason code*, *table number* and *function code* values can be looked up in "Collector table manager diagnostics" on page 314.

Module: CIUCINB2

Destination

CINT TD queue

CIU3315S File *filename* is full

Explanation: Transaction CINB received a NOSPACE response when issuing EXEC CICS WRITE for VSAM dependency file *filename*. The file has filled up.

System action: The transaction is terminated by abending IUYJ and the Collector is stopped.

User response: Allocate more space to the file and rerun the Collector.

Module: CIUCINB2

Destination

CINT TD queue

CIU3316S Unexpected resource type *resourcetype*

Explanation: Transaction CINB received an unrecognized resource type from another Collector component.

System action: The Collector is stopped.

User response: Contact your CICS IA support representative.

Module: CIUCIND

Destination

Terminal end user.

| **CIU3320I CIUCPOOL: Limit of cells is exceeded.**
| **NNNNN records rejected**

| **Explanation:** NNNNN records were rejected because
| the limit for the number of simultaneously selected
| CPOOL cells is exceeded.

| **System action:** The counter of the rejected records is set to zero.

| **User response:** None.

| **Module:** CIUA120C, CIUA130C, CIUCINB2

| **Destination**

| CINT TD queue

CIU3321I Performance Monitoring is activated.

Explanation: The CINC transaction activated the CICS monitoring for performance data.

System action: CINC transaction activates the CICS monitoring (Monitoring Domain) for performance data.

User response: None.

Module: CIUACM61

Destination

CINT TDQ.

CIU3322I Performance Monitoring is deactivated

Explanation: The CINC transaction deactivated CICS monitoring for performance data.

System action: CINC transaction deactivates the CICS monitoring (Monitoring Domain) for performance data.

User response: None.

Module: CIUACM71

Destination

CINT TDQ

CIU3323I The TCB switch counter data is not available

Explanation: The CICS Monitoring Domain stopped during the execution of the command flow instance and therefore the collected data contains no TCB switch counters.

System action: The TCB switch counters are set to "-1."

User response: None.

Module: CIUACM71

Destination

CINT TDQ.

CIU3401I Master System trace flag is set ON

Explanation: CICS IA sets Master System trace flag ON.

System action: None.

User response: None.

Module: CIUA105C, CIUA110C, CIUA140C

Destination

CINT TD queue

CIU3402I Master User trace flag is set ON

Explanation: CICS IA sets Master User trace flag ON.

System action: None.

User response: None.

Module: CIUA105C, CIUA110C, CIUA140C

Destination

CINT TD queue

CIU3405I Master System trace flag is restored OFF

Explanation: Master System trace flag is restored to OFF after CICS IA is stopped or paused, when the Restore Master Trace value is YES.

System action: None.

User response: None.

Module: CIUA120C, CIUA130C

Destination

CINT TD queue

CIU3406I Master User trace flag is restored OFF

Explanation: Master User trace flag is restored to OFF after CICS IA is stopped or paused, when the Restore Master Trace value is YES.

System action: None.

User response: None.

Module: CIUA120C, CIUA130C

Destination

CINT TD queue

CIU3407I Error at setting Master trace flag. Response code: eibresp

Explanation: CICS IA was unable to set master trace flag.

System action: None.

User response: None.

Module: CIUA105C, CIUA110C, CIUA120C, CIUA130C, CIUA140C

Destination

CINT TD queue

CIU3408I **ENTER TRACENUM for program name failed**
RESP=eibresp . HL_TRACE forced to N

Explanation: Program **program name** received an invalid response when issuing the EXEC CICS ENTER TRACENUM command. The response is in eibresp.

System action: The value of HL_TRACE is forced to N(No), and High level trace is stopped.

User response: Check the response code. Check Master User trace flag for value INVREQ: may be it was changed to OFF manually or by another program.

Module: CIUA105C, CIUA110C, CIUA120C, CIUA130C, CIUA140C, CIUA180C, CIUCINB1, CIUCINB2, CIUCINBE, CIUCINTE

Destination

CINT TD queue

CIU4100S **Function call failed FUNCTION=function code**
REASON=reason code TABLE=table number

Explanation: One of the Collector exit programs received an unexpected response when issuing a call to the table manager, CIUTABM. Note that the message is issued by transaction CINB on behalf of the exit program.

System action: The CINB transaction is terminated with abend code IUXA and the Collector is stopped.

User response: The *reason code*, *table number* and *function code* values can be looked up in “Collector table manager diagnostics” on page 314.

Module: CIUCINB1

Destination

CINT TD queue

CIU4200S **Dataspace is full**

Explanation: Transaction CINT or a Collector exit program received a reason code of AUTM_NO_STORAGE when issuing a call to the table manager, CIUTABM, to either create a new table or add an element to a table. The data space has filled up. Note that if this situation was encountered by a Collector exit program, the message is issued by transaction CINB on its behalf.

System action: The CINT or CINB transaction is terminated by abending IUXB and the Collector is stopped.

User response: Increase the data space size using the CINT options and rerun the Collector.

Module: CIUA110C, CIUA140C, CIUCINB1

Destination

CINT TD queue

CIU4300E **Interaction with Natural failed**
REASON=reason

Explanation: The CICS IA interface to Natural has detected an inconsistent (recoverable) error during interaction with Natural SYSRDC. The following list gives possible reasons:

- SILOST: Natural session initialization event is lost.
- STLOST: Natural session termination event is lost.
- PLLOST: Program load event is lost.
- EWAOVER: Natural SYSRDC exit work area overflow, not enough storage for all the Natural program names.
- EWAMISM: Natural SYSRDC exit work area mismatch, possibly the exit work area overflowed previously.
- EWAEXHST: Natural SYSRDC exit work area is exhausted, possibly the exit work area overflowed previously.
- NNLOVER: Natural Name List overflow, not enough storage for all the Natural sessions.
- NNLMISM: Natural Name List mismatch. It is possible that the list overflowed previously.

System action: The interface tries to recover and continue interaction with Natural SYSRDC, but it is possible that not all Natural programs will be identified correctly in the data collected.

User response: The following list gives possible actions based on the reason:

- SILOST: None.
- STLOST: None.
- PLLOST: None.
- EWAOVER: Change the RDCEXIT Natural profile parameter for CIURDCX1 specifying a larger exit work area.
- EWAMISM: None.
- EWAEXHST: None.
- NNLOVER: Recreate the Natural Name List specifying a larger number of Natural sessions.
- NNLMISM: None.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU4307S **Interaction with Natural failed**
REASON=reason

Explanation: The CICS IA interface to Natural has detected an unexpected (unrecoverable) error during interaction with Natural SYSRDC. The following is the list of possible reasons:

- UNXSID: Unexpected Natural session identifier.
- INVNNL: Invalid Natural Name List.

System action: The interface is terminated by with abend code IUZ7 but the Collector is not stopped.

User response: Contact your CICS system support team.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU4308S **CICS command data failed RESP=eibresp RESP2=eibresp2 RCODE=eibrcode**

Explanation: The CICS IA interface to Natural received an invalid response when issuing command EXEC CICS. The response is in *eibresp*, *eibresp2* and *eibrcode*. Other data, if present, might give the object operated on by the command.

System action: The interface is terminated by abending IUZ8 but the Collector is not stopped.

User response: For further details of the exception eibresp refer to the command in the *CICS Application Programming Reference manual* or the *CICS System Programming Reference manual*. For further information on how to determine system problems refer to the *CICS Problem Determination Guide*.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU4309S **Interaction with Natural has abended abend code**

Explanation: This message is issued when the HANDLE ABEND exit program of the CICS IA interface to Natural is driven to handle an abend that occurred within the interface. The abend code is given by *abend code*.

System action: The interface is terminated by abending IUZ9 but the Collector is not stopped.

User response: The abend was issued by CICS, for example, ASRA, so refer to the *CICS Messages and Codes*.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU5000S **Function call num is invalid for module**

Explanation: A module of the Reporter has been called with an invalid function number, indicating an internal logic error in the Reporter.

System action: The Reporter is terminated.

User response: Contact IBM support.

Module: CIUREPFM, CIUREPPM

Destination

Console.

CIU5001E **filename control record not found for applid applid. Run canceled**

Explanation: No control record has been found in CIUCNTL for the CICS APPLID.

System action: The Reporter is terminated.

User response: Check that the correct files are being input to the Reporter. Check that the Applid you have selected is correct.

Module: CIUREP

Destination

Console.

CIU5003E **Some action failed. Return/ABEND code=return reason=reason**

Explanation: A batch program was unable to complete because *some action* failed.

System action: The batch program is terminated.

User response: If *some action* is opening a VSAM file then check the return code and reason in the VSAM messages and codes manual to determine the cause of the error. If *some action* is opening a non-VSAM file then the most likely cause of this message will be missing or incorrect filenames in the JCL to run the job. Correct the JCL and rerun the job. If *some action* is GETMAIN or IEWBUFF then the most likely cause of this message will be shortage of virtual storage. Change your JCL to allow more and rerun the job.

Module: CIUREPFM, CIULMS, CIUCSS

Destination

Console.

CIU5004E **GENCB failed for filename CB=control block RC=return code REASON=reason code**

Explanation: The generation of a VSAM *control block* failed for Reporter file *filename*.

System action: The Reporter is terminated.

User response: The *return code* and *reason code* are as returned by VSAM in GPR 15 and GPR 0 respectively. Check the VSAM messages and codes manual to determine the cause of the error. Correct the problem and rerun the job.

Module: CIUREPFM

Destination

Console.

CIU5005S File number *filenum* is invalid

Explanation: The file manager module, CIUREPFM, used by the Reporter has been called with an invalid file number *filenum*.

System action: The Reporter is terminated.

User response: Rerun the job.

Module: CIUREPFM

Destination

Console.

CIU5006S Attempt to *actionfilename*. File is type

Explanation: The Reporter attempted to read from the output file or write to the input file specified by *filename*.

System action: The Reporter is terminated.

User response: Rerun the job.

Module: CIUREPFM

Destination

Console.

CIU5007S RPL number *rplnum* is invalid for *filename*

Explanation: The RPL number *rplnum* is invalid for Reporter file *filename*.

System action: The Reporter is terminated.

User response: Rerun the job.

Module: CIUREPFM

Destination

Console.

CIU5008S Table number *table number* is invalid

Explanation: A request by the Reporter to read the table from the dependency data file was being processed but *table number* was not valid.

System action: The Reporter is terminated.

User response: Rerun the job.

Module: CIUREPFM

Destination

Console.

| **CIU5012E** Invalid *PARM* specified. Program terminated.

| **Explanation:** A syntax error was detected in the specified *PARM* information when invoking the Scanner.

| **System action:** The Scanner is terminated with RC=8.

| **User response:** Correct the *PARM* information and run the job again.

| **Module:** CIULMS, CIUCSS

| **Destination**

| Console.

CIU5015E Invalid *{MATCH | STATE | CONTEXT}* value specified. Correct and rerun

Explanation: When invoking the Builder, a *PARM* field has been specified on the EXEC that contains an invalid value for the keyword given.

Keyword	Allowed values
MATCH	LUNAME and USERID
STATE	ACTIVE and DORMANT
CONTEXT	plexname, 1 through 8 characters

System action: The Builder is terminated.

User response: Correct the *PARM* information and rerun the job. This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5016E *DSPSIZE* value is not numeric. Correct and rerun

Explanation: When invoking the Builder, a *PARM* field has been specified on the EXEC that contains an invalid value for keyword *DSPSIZE*. A character other than a digit in the range 0 through 9 was encountered. The value must be an integer in the range 2 through 2000.

System action: The Builder is terminated.

User response: Correct the *PARM* information and rerun the job. This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5017E *DSPSIZE* is invalid. It must be between 2 and 2000

Explanation: When invoking the Builder, a *PARM* field has been specified on the EXEC that contains an invalid value for keyword *DSPSIZE*. The value must be an integer in the range 2 through 2000.

System action: The Builder is terminated.

User response: Correct the PARM information and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5018E Load of CIUTABM has failed AC *abcode*
RC *reason_code*

Explanation: The Builder attempted to load the table manager module, CIUTABM, but the MVS LOAD macro failed. The *abcode* is returned in GPR 1 and is the abend code that would have resulted had the task abended. The *reason_code* is returned in GPR 15 and is the reason code associated with the abend.

System action: The Builder is terminated.

User response: The MVS *abcode* and *reason_code* indicate the cause of the error. Correct the problem and rerun the job.

Note: This message can not be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5019E Dataspace too large - no storage available

Explanation: The Builder received a response from the table manager, CIUTABM, reporting that it was unable to obtain the amount of storage requested for the MVS data space because the MVS real storage manager does not have enough resources.

System action: The Builder is terminated.

User response: Decrease the data space size specified on the PARM field of the EXEC statement in the job, and rerun the job. This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5020E Dataspace too large – IEFUSI limit reached

Explanation: The Builder received a response from the table manager, CIUTABM, that it was unable to obtain the amount of storage requested for the MVS data space, because MVS exit IEFUSI has imposed a limit on address space size.

System action: The Builder is terminated.

User response: Either decrease the data space size specified on the PARM field of the EXEC statement in the job, or ask your MVS system programmer to increase the IEFUSI limit. Rerun the job.

Note: This message cannot be changed with the message editing utility

Module: CIUBLD

Destination

Console

CIU5021E Create dataspace failed REASON
reason_code ERROR *error_code*

Explanation: The Builder received an invalid response when issuing a call to the table manager, CIUTABM, to create the MVS data space.

System action: The Builder is terminated.

User response: See the *CICS Transaction Affinities Utility Guide* for the meaning of the reason code. If it is AUTM_DSPSERV_CREATE_ERROR, *error_code* is the value of GPR 0 after the MVS DSPSERV CREATE call. If it is AUTM_ALESERV_ADD_ERROR, *error_code* is the value of GPR 15 after the MVS ALESERV ADD call. Use the appropriate MVS manual to find out the meaning of the error code.

Note: This message cannot be changed with the message editing utility

Module: CIUBLD

Destination

Console

CIU5022E Create table failed REASON *reason_code*
TABLE *table_number*

Explanation: The Builder received an invalid response when issuing a create table call to the table manager, CIUTABM, for table *table_number*.

System action: The Builder is terminated.

User response: See the *CICS Transaction Affinities Utility Guide* for the meaning of the reason code and the table number.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5023E Keyword *keyword* is invalid or unexpected

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a keyword it did not recognize, or a keyword that occurred in an unexpected place.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job. A likely cause of this message is the omission of the terminating semi-colon from the preceding statement.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5024E Keyword *keyword* is missing

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a statement in which the required keyword *keyword* was missing.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5025E A value of *value* is invalid for keyword *keyword*

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered an invalid value for keyword *keyword*.

System action: The Builder skips to the next keyword and continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5026E Invalid CREATE type

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a CREATE statement. The only keywords that are allowed to follow immediately after the CREATE keyword are TRANGRP and DTRINGRP.

System action: The Builder skips to the next input statement and continues, but will terminate when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5027E Incorrect number of brackets

Explanation: When reading statements from its REPGRPS input stream the Builder encountered a keyword for which a corresponding value was expected. This value is invalid because it:

- Does not start with a bracket
- Does not end with a bracket
- Contains a bracket in the middle
- Spans input lines.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5028E Missing semicolon

Explanation: When reading statements from its REPGRPS input stream the Builder encountered the end of the input in the middle of a statement, implying

that a terminating semicolon is missing from the last statement.

System action: The Builder is terminated.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5029E Keyword *keyword* is duplicated

Explanation: When reading statements from its REPGRPS input stream the Builder encountered a keyword that occurred more than once in the same statement. Duplicate keywords are not allowed.

System action: The Builder continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5030E No valid statements in REPGRPS – processing terminated

Explanation: When reading statements from its REPGRPS input stream the Builder did not find a single complete statement.

System action: The Builder is terminated.

User response: The most likely cause of this message is an empty input file. Correct the problem and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

Console

CIU5031E CIUTABM error *function* element TABLE *table_number* REASON *reason_code* MODULE *progname*

Explanation: Builder module *progname* received an

unexpected response when issuing a call to the table manager, CIUTABM.

System action: The Builder is terminated.

User response: The *function* is the operation being performed. See the *CICS Transaction Affinities Utility Guide* for the meaning of the reason code and the table number.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR, CIUBLDOT

Destination

Console

CIU5032E No HEADER record found – statement ignored

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a CREATE statement. However, no HEADER statement had been encountered first. The HEADER statement is mandatory and must be the first statement in each data set in the REPGRPS concatenation.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Ensure that each data set in the REPGRPS concatenation has a HEADER statement as the first statement in the data set. Correct the problem and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

SYSPRINT

CIU5033E Duplicate TRANGRP name

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a CREATE TRANGRP statement. However, the Trangroup name supplied in the statement is not unique within the current input data set. Duplicate Trangroup names are not allowed.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Ensure that each CREATE TRANGRP statement within the data set specifies a unique Trangroup name. If this is already the case, the probable cause of this message is that the HEADER statement is missing from the data set. Correct the problem and rerun the job.

CIU5034E • CIU5038E

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

SYSPRINT

CIU5034E Transaction group has not been previously created

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a CREATE DTRINGRP statement. However, no corresponding valid CREATE TRANGRP statement for the Trangroup in question was encountered.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Either the CREATE TRANGRP statement was missing or was in error. Correct the problem and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

SYSPRINT

CIU5035W Dependency data may be incomplete because of *abend type abend*

Explanation: The control record on dependency control file CIUCNTL indicates that the Collector did not stop cleanly. Either CICS crashed or the Collector abended, as indicated by *abend type*. If the termination occurred during a Collector save, it is possible that the data on the dependency file might be incomplete.

System action: The Reporter continues.

User response: If incomplete data is a significant problem to you, rerun the Collector to ensure a complete set of data.

Note: This message cannot be changed with the message editing utility.

Module: CIUREP

Destination

Console

CIU5036E Dataspace is full

Explanation: A Builder module received a reason code of AUTM_NO_STORAGE when issuing a call to the table manager, CIUTABM to add an element to a table. The data space is full.

System action: The Builder is terminated.

User response: Increase the data space size specified on the PARM field of the EXEC statement in the job, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

Console

CIU5037E No valid transaction IDs in REPGRPS – processing terminated

Explanation: When reading statements from its REPGRPS input stream, the Builder did not find a single valid CREATE DTRINGRP statement.

System action: The Builder is terminated.

User response: The most likely cause of this message is an input stream that contains valid CREATE TRANGRP statements, but no CREATE DTRINGRP statements. Correct the problem and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

Console

CIU5038E Invalid AFFLIFE for AFFINITY

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a CREATE TRANGRP statement. However, the value specified for AFFLIFE is not one of those permitted given the value specified for AFFINITY.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Refer to *CICS Problem Determination Guide* to correct the statement, and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDMR

Destination

SYSPRINT

CIU5039E PARM keyword is duplicated. Correct and rerun

Explanation: When invoking the Builder or Reporter, a PARM field has been specified on the EXEC that contains a duplicate keyword. Duplicate keywords are not allowed.

System action: The Builder or Reporter is terminated.

User response: Correct the PARM information and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLD

Destination

Console

CIU5040E Invalid REMOVE type

Explanation: When reading statements from its REPGRPS input stream, the Builder encountered a REMOVE statement. The only keyword allowed to follow immediately after the REMOVE keyword is TRANGRP.

System action: The Builder skips to the next input statement and continues, but terminates when the end of the input is encountered.

User response: Refer to the *CICS Transaction Affinities Utility Guide* to correct the statement, and rerun the job. Alternatively, comment out the statement.

Note: This message cannot be changed with the message editing utility.

Module: CIUBLDIN

Destination

SYSPRINT

CIU5041E Report type is not recognized — processing terminated

Explanation: The user has entered an invalid report type.

System action: The Reporter is terminated.

User response: Enter a valid report type: CICS, MQ, DB2, IMS, or ALL.

Note: This message cannot be changed with the message editing utility.

Module: CIUREP

Destination

Console

CIU5042I Error reading Applid input – processing terminated

Explanation: When invoking the Reporter, a PARM field has been specified on the EXEC that contains an invalid keyword. The only allowable keyword is WORSEN.

System action: The Reporter is terminated.

User response: Correct the PARM information and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUREP

Destination

Console

CIU5043I Error reading CIUCNTL file – processing terminated

Explanation: When invoking the Reporter, a PARM field has been specified on the EXEC that contains an invalid value for the keyword given.

Keyword	Allowed values
WORSEN	YES and NO

System action: The Reporter is terminated.

User response: Correct the PARM information and rerun the job.

Note: This message cannot be changed with the message editing utility.

Module: CIUREP

Destination

Console

CIU5044E DB2 table error on *tablename* SQL code *code*

Explanation: SQL error code *code* occurred while querying DB2 table *tablename*.

System action: Program terminates normally but some expected output might be missing.

User response: Ensure that CICS IA is properly installed. Contact your system support group.

Module: CIULMS

Destination

Console

CIU5050E **OPTION ERROR:** *modulename* A VALUE OF *optionvalue* IS INVALID FOR OPTION *optionname*.

Explanation: *optionvalue* is not valid for the option specified by the *optionname*. *modulename* is the name of the module issuing the message.

System action: The job is terminated.

User response: Correct the value of the specified option and rerun the job.

Module: CIUNTSQ2

Destination

SYSOUT

CIU5051E *modulename* ERROR ON OPEN OF CIUOPTS

Explanation: An error was detected when the Threadsafe Analysis report attempted to open the option file. *modulename* is the name of the module issuing the message.

System action: The job is terminated.

User response: Verify that the option file CIUOPTS is present and rerun the job.

Module: CIUNTSQ2

Destination

SYSOUT

CIU6002E **Bad parameter: xxx**

| **Explanation:**

| Jobs CIUUPDB, CIUUPDBN, CIUUPDB1, CIUUPDB2, CIUUPDB3, CIUUPDB4 have an incorrect value in the PARM parameter in STEP040, STEP045, STEPN40, STEP090, STEP130 and STEP170. It must be UPD to update timestamps or NOPARM to not update timestamps.

| Jobs CIUAFFLX and CIUAFFRD have an incorrect value in the PARM parameter in STEP010.

System action: The job is terminated.

User response: Correct the parameter.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUU055, CIUU05N, CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUU056.

| **Destination**

| Console.

CIU6003I **Last use timestamps will not be updated**

Explanation: The last observed timestamps will not be updated on the database.

System action: Last observed timestamps are not updated.

User response: None.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUU055, CIUU05N, CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUU056

| **Destination**

| Console.

CIU6004I **Last use timestamps will be updated**

Explanation: The last observed timestamps will be updated on the database. Note that this action will probably increase the database update time.

System action: Last observed timestamps are updated.

User response: None.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUU055, CIUU056, CIUU05N, CIUAFFL1, CIUAFFL2, CIUAFFL3

| **Destination**

| Console.

CIU6005I **Number of new rows added to *tablename* = *nnnn***

Explanation: The number of rows that have been inserted into the DB2 table specified by *tablename*.

System action: Program terminates normally after issuing this message.

User response: None.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUU055, CIUUREG, CIUU05N, CIUAFFL1, CIUAFFL2, CIUAFFL3

| **Destination**

| Console.

CIU6006I **Number of existing rows in *tablename* = *nnnn***

Explanation: The number of rows that already exist in the table specified by *tablename*.

System action: Program terminates normally after issuing this message.

User response: None.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUU056, CIUUREG

| **Destination**

| Console.

CIU6007I **Number of rows updated in** *tablename* =
 nnnn

Explanation: The number of rows that were updated with the last observed timestamp in the table specified by *tablename*.

System action: Program terminates normally after issuing this message.

User response: None.

Module: CIUU050, CIUU051, CIUU052, CIUU053,
| CIUU055, CIUU05N, CIUAFFL1, CIUAFFL2,
| CIUAFFL3, CIUU056, CIUUREG

| **Destination**

| Console.

CIU6008E **SQL error:** *xxxxxxxx, yyyyyyyyyy*

Explanation: SQL error occurred, where:

- *xxxxxxxx* is the name of the module that detected the error.
- *yyyyyyyyyy* contains diagnostic information about the error.

System action: The program terminates with return code 12.

User response: The diagnostic information shows the SQL return code. Refer to the *DB2 Messages and Codes* manual.

Module: CIUU050, CIUU051, CIUU052, CIUU053,
| CIUU055, CIUUREG, CIUAFFR2, CIUAPEXT,
| CIUCFUPD, CIUU056B, CIUU056C, CIUU056D,
| CIUU056E, CIUU056F, CIUU056G, CIUU056, CIUU05N,
| CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUNTSQ2

| **Destination**

| Console.

CIU6010E **File error:** *xxxxxxxx, yyyyyyyyyy*

Explanation: An I/O error occurred, where:

- *xxxxxxxx* is the name of the program
- *yyyyyyyyyy* is diagnostic information

System action: The program terminates with return code 12.

User response: The diagnostic information shows the I/O return code. Refer to the *IBM COBOL for MVS and VM Language Reference* manual.

Module: CIUU044, CIUU050, CIUU051, CIUU052,
| CIUU053, CIUU055, CIUNTSQ2, CIUAPEXT,
| CIUAPPRS, CIUAFFR2, CIUU056, CIUU05N,

CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUUREG,
| CIUMIGVF, CIUCFUPD

| **Destination**

| Console.

CIU6011I ****** QSAM OUTPUT STATISTICS FOR**
| **TABLE** *tablename* ********

Explanation: This message is used as a section header when reporting the update status of the DB2 table specified by *tablename*.

System action: None.

User response: None.

Module: CIUU056A, CIUU056B, CIUU056C,
| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

CIU6012I -----

Explanation: This message is used as a spacer between messages.

System action: None.

User response: None.

Module: CIUU056A, CIUU056B, CIUU056C,
| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

CIU6013I **Number of processed records:**

Explanation: The number of processed records in the table. Message CIU6011I specifies the table updated.

System action: The program terminates normally after issuing this message.

User response: None.

Module: CIUU056A, CIUU056B, CIUU056C,
| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

CIU6014I **Number of existing rows:**

Explanation: The number of rows that already exist in the table. Message CIU6022I specifies the table updated.

System action: Program terminates normally after issuing this message.

User response: None.

Module: CIUU056A, CIUU056B, CIUU056C,

| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

CIU6015I Number of rows updated:

Explanation: The number of rows updated with the last observed timestamp. Message CIU6022I specifies the table updated.

System action: Program terminates normally after issuing this message.

User response: None

| **Module:** CIUU056A, CIUU056B, CIUU056C,
| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

CIU6016E CIU_THREADSAFE_CMD table is empty.

Explanation: The DB2 table CIU_THREADSAFE_CMD does not contain any data.

System action: Program terminates with a nonzero return code after issuing this message.

User response: Run job CIUTSLOD to load the CIU_THREADSAFE_CMD table and rerun job CIUJTSQ2.

Module: CIUNTSQ2

| **Destination**

| Console.

| **CIU6017I QSAM CSV files will be produced**

| **Explanation:** QSAM CSV files will be produced.

| **System action:** The program terminates normally after issuing this message.

| **User response:** None.

| **Module:** CIUU056

| **Destination**

| Console.

**CIU6018E QSAM error, xxxxxxxx, yyyyyyyy.
 X=program name; y=diagnostic information**

Explanation: A QSAM error occurred

System action: The program terminates with return code 8.

User response: The diagnostic information shows the QSAM return code. For further information, refer to the

IBM COBOL for MVS and VM Language Reference Manual.

Module: CIUU056, CIUU044

| **Destination**

| Console.

CIU6019I Journal records read = nnnnnnnnn

Explanation: nnnnnnnnn is the number of journal records read from the IA log stream data sets.

System action: None.

User response: None.

Module: CIUU044

| **Destination**

| Console.

CIU6020I Trace records written = nnnnnnnnn

Explanation: nnnnnnnnn is the number of command trace records written to the sequential data set.

System action: None.

User response: None.

Module: CIUU044

| **Destination**

| Console.

| **CIU6021W No rows selected for processing from
 tablename**

| **Explanation:** No rows selected for processing from
| tablename.

| **System action:** None.

| **User response:** None.

| **Module:** CIUU050, CIUU051, CIUU052, CIUU053,
| CIUU055, CIUU05N

| **Destination**

| Console.

| **CIU6022I **** UPDATE STATUS FOR TABLE
 tablename ******

| **Explanation:** This message is used as a section header
| when reporting the update status of the DB2 table
| specified by tablename.

| **System action:** None.

| **User response:** None.

| **Module:** CIUU056A, CIUU056B, CIUU056C,
| CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

| **CIU6023I** **Number of new rows added:**

| **Explanation:** The number of new rows added in the table. Message CIU6022I specifies the table updated.

| **System action:** The program terminates normally after issuing this message.

| **User response:** None.

| **Module:** CIUU056A, CIUU056B, CIUU056C, CIUU056D, CIUU056E, CIUU056F, CIUU056G.

| **Destination**

| Console.

| **CIU6024E** **SQLERRMC: sqlerrmc**

| **Explanation:** The SQLERRMC information if the SQLCODE is nonzero.

| **System action:** The program terminates with code 8.

| **User response:** Analyze the error message, correct information if possible and rerun the program.

| **Module:** CIUAFFL1, CIUAFFL2, CIUAFFL3, CIUU050, CIUU052, CIUU053, CIUU055, CIUU056A, CIUU056B, CIUU056C, CIUU056D, CIUU056E, CIUU056F, CIUU056G, CIUU05N, CIUUREG

| **Destination**

| Console.

CIU6030I **CONTROL1 record added to control file**

Explanation: A new CONTROL1 type record is written to the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6031I **CONTROL1 record already exists**

Explanation: CONTROL1 type record already exists in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6032I **Defaults record added to control file**

Explanation: A new DEFAULTS type record is written to the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6033I **DEFAULTS record already exists**

Explanation: A DEFAULTS type record already exists in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6034I **APPLID region record added to control file**

Explanation: A new REGION type record for the supplied APPLID is written to the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6035I **APPLID region record already exists**

Explanation: A REGION type record for the supplied APPLID already exists in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

| **Destination**

| Console.

CIU6036I **Number of regions added =number**

Explanation: The number of REGION type records added to the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

Destination

Console.

CIU6037I Number of existing regions =*number*

Explanation: The number of REGION type records already existing in the CIUCNTL control file.

System action: None.

User response: None.

Module: CIUCFUPD

Destination

Console.

CIU6038I Number of error records =*number*

Explanation: The number of //REGION DD card entries in error.

System action: None.

User response: None.

Module: CIUCFUPD

Destination

Console.

CIU6039W No input records found for //REGIONS

Explanation: No region input is added to the //REGIONS DD card in the sample CIUJCUPD.

System action: None.

User response: Review the //REGIONS DD card in sample CIUJCUPD.

Module: CIUCFUPD

Destination

Console.

CIU6040E Input record error – add region(*aaaaaaaa,ssss*)

Explanation: An incorrect entry is added to the //REGIONS DD card.

System action: None.

User response: Review the entry in the //REGIONS DD card. Ensure that the CICS APPLID, *aaaaaaaa*, and the SYSID, *ssss*, are correct.

Module: CIUCFUPD

Destination

Console.

CIU6041E CICS APPLID must be 1 to 8 characters

Explanation: An incorrect CICS APPLID is entered in the //REGIONS DD card.

System action: None.

User response: Review the entry in the //REGIONS DD card. Ensure that the CICS APPLID, *aaaaaaaa*, and the SYSID, *ssss*, are correct.

Module: CIUCFUPD

Destination

Console.

CIU6042E CICS SYSID must be 1 to 4 characters

Explanation: An incorrect CICS SYSID is entered in the //REGIONS DD card.

System action: None.

User response: Review the entry in the //REGIONS DD card. Ensure that the CICS APPLID, *aaaaaaaa*, and the SYSID, *ssss*, are correct.

Module: CIUCFUPD

Destination

Console.

CIU6043W No input records found for //CIUMIGXT

Explanation: No input records found for //CIUMIGXT.

System action: The program terminates with return code 8.

User response: Review the entry in the //CIUMIGXT DD card.

Module: CIUAPPRS

Destination

Console.

CIU6044I Parse OK - Number of APPLS added =

Explanation: The input data is right.

System action: None.

User response: None.

Module: CIUAPPRS

Destination

Console.

| **CIU6045I** **Parse failed - XML-EVENT: XML-TEXT:**
| **Explanation:** The input data is bad.
| **System action:** The program terminates with return
| code 8.
| **User response:** Test the input data and rerun the
| program.
| **Module:** CIUAPPRS
| **Destination**
| Console.

| **CIU6046I** **Number of duplicate rows fixed in**
| *tablename = count*
| **Explanation:** An information message.
| **System action:** None.
| **User response:** None.
| **Module:** CIUU052
| **Destination**
| Console.

| **CIU6047I** **Number of applications migrated =**
| **Explanation:** An information message.
| **System action:** None.
| **User response:** None.
| **Module:** CIUAPEXT
| **Destination**
| Console.

| **CIU6048E** **Length of APPLIC code > 8**
| **Explanation:** The input data is bad.
| **System action:** The program terminates with return
| code 8.
| **User response:** Test the input data and rerun the
| program.
| **Module:** CIUAPPRS
| **Destination**
| Console.

| **CIU6049E** **Length of APPLIC name > 50**
| **Explanation:** The input data is bad.
| **System action:** The program terminates with return
| code 8.
| **User response:** Test the input data and rerun the
| program.

| **Module:** CIUAPPRS
| **Destination**
| Console.

| **CIU6050E** **Length of APPLIC tran > 4**
| **Explanation:** The input data is bad.
| **System action:** The program terminates with return
| code 8.
| **User response:** Test the input data and rerun the
| program.

| **Module:** CIUAPPRS
| **Destination**
| Console.

| **CIU6051E** **Length of APPLIC program > 8**
| **Explanation:** The input data is bad.
| **System action:** The program terminates with return
| code 8.
| **User response:** Test the input data and rerun the
| program.

| **Module:** CIUAPPRS
| **Destination**
| Console.

| **CIU7000I** **5655-U86 (C) Copyright IBM Corp.**
| **2001,2011**
| **Explanation:** The CICS IA copyright message.
| **System action:** None.
| **User response:** None.

| **Module:** CIUA000C, CIUA400C, CIUACM10
| **Destination**
| Terminal end user.

| **CIU7001S** **An irrecoverable error has occurred in**
| *module module.*
| **Explanation:** An error has occurred in the *module* part
| of the request processing. The *module* values are:
| • START
| • STOP
| • STATISTICS
| • WEB SERVICE

| **System action:** The XML response containing the
| message is sent to the user.
| **User response:** Refer to the CIU7047S message, and
| then to CIU2201S.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

CIU7003I Enter a valid option

Explanation: You entered a value that is invalid in this context.

System action: None.

User response: Enter a valid value.

Module: CIUA400C

Destination

Terminal end user.

CIU7027I No regions defined to CICS IA

Explanation: No CICS regions have been defined to the Collector.

System action: None.

User response: On the Collector's Region Configuration Menu, specify at least one CICS region to be monitored.

Module: CIUA100C, CIUA200C

Destination

Terminal end user.

CIU7028W Invalid action code.

Explanation: You entered an invalid value. Valid values are numbers in the range 1 through 5.

System action: None.

User response: Enter a numeric value in the range 1 through 5.

Module: CIUA100C, CIUA200C

Destination

Terminal end user.

CIU7029I Action processed successfully.

Explanation: The specified action has completed successfully.

System action: None.

User response: None.

Module: CIUA100C, CIUA200C

Destination

Terminal end user.

CIU7030I Start/Stop request cancelled.

Explanation: You entered F12 to cancel the requested action.

System action: The start or stop action is not performed.

User response: None.

Module: CIUA100C

Destination

Terminal end user.

CIU7031E Connection to remote region not acquired

Explanation: CICS IA could not acquire a connection to a remote CICS region.

System action: None.

User response: Ensure that CICS IA is properly installed. Contact your system support group.

Module: CIUA100C

Destination

Terminal end user.

CIU7032E Connection to remote region not found

Explanation: CICS IA could not find a CONNECTION definition for the remote region.

System action: None.

User response: Ensure that CICS IA is properly installed. Contact your system support group.

Module: CIUA100C

Destination

Terminal end user.

CIU7033I Enter new Applid and Sysid

Explanation: You have requested to add or copy a new region and have not supplied the required information.

System action: None.

User response: Enter the VTAM application identifier (Applid) and the CICS system identifier (Sysid) of the new CICS region being defined to CICS IA.

Module: CIUA200C

Destination

Terminal end user.

CIU7034I New Applid already exists

Explanation: You have requested to add or copy a new region but the Applid you have entered is already defined to CICS IA.

System action: None.

User response: Check that you have entered the correct Applid and, if required, try again.

Module: CIUA200C

Destination

Terminal end user.

CIU7035I Press Enter to confirm delete or PF12 to cancel

Explanation: You have requested to delete a CICS region from CICS IA.

System action: None.

User response: Press Enter to confirm deletion of this region, or PF12 to cancel the delete.

Module: CIUA200C

Destination

Terminal end user.

CIU7036I Request cancelled

Explanation: You have entered F12 to cancel the delete request.

System action: None.

User response: None.

Module: CIUA200C

Destination

Terminal end user.

CIU7037I State must be STOPPED before you can delete

Explanation: You have chosen to delete a region from CICS IA, but the Collector is still running on that region.

System action: None.

User response: Stop the Collector on the region you want to delete and try again.

Module: CIUA200C

Destination

Terminal end user.

CIU7038I Options 1-3 not valid for single region file

Explanation: You have tried to add, copy, or delete a region in the Collector, but CICS IA is not configured to support multiple regions in one VSAM file.

System action: None.

User response: Review how CICS IA is installed and customized.

Module: CIUA200C

Destination

Terminal end user.

CIU7039I Press Enter to confirm action or PF12 to cancel

Explanation: You have chosen to start or stop the Collector on a selected region.

System action: None.

User response: Press Enter to confirm this action, or PF12 to cancel the action.

Module: CIUA100C

Destination

Terminal end user.

CIU7041I CICS IA not installed in target region

Explanation: CICS IA is not correctly installed in the target region.

System action: The region state is set to UNCONNECTED.

User response: Ensure that CICS IA is installed in the target region if required.

Module: CIUA100C

Destination

Terminal end user.

CIU7042I Option number not valid for item

Explanation: The option number entered is not valid for the list item it has been entered against.

System action: The CINT options will not be accepted unless all of the input is correct.

User response: Correct the invalid input.

Module: CIUA100C, CIUA200C

Destination

Terminal end user.

CIU7043I Maximum of 200 records in control file

Explanation: CICS IA is limited to 200 records in the control file.

System action: Attempts to add more region records are rejected.

User response: Remove any unnecessary region records to allow space for the ones you need.

Module: CIUA200C

Destination

Terminal end user.

| **CIU7044I** CINC Action processed successfully.

| **Explanation:** The action done via Web Service was processed successfully.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** None.

| **Module:** CIUATM03

| **Destination**

| Client.

| **CIU7045E** CINC Incorrect user *userid* entry.

| **Explanation:** The user record in the Control file has an incorrect format.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** Contact your CICS system support person.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

| **CIU7046E** CINC user *userid* entry not found.

| **Explanation:** The user record was not found in the Control file.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** Contact your CICS system support person.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

| **CIU7047S** CINC Unknown request has been received.

| **Explanation:** Web Service received an unknown request from the client.

| **System action:** Program terminates with the IUXH abend code.

| **User response:** Contact your CICS system support person.

| **Module:** CIUATM03

| **Destination**

| CINT TD queue

| **CIU7048E** CINC CONTROL1 record is not found in Control file.

| **Explanation:** The CONTROL1 record was not found in the Control file.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** Contact your CICS system support person.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

| **CIU7049E** CINC CONTROL1 record in Control file is corrupted.

| **Explanation:** The CONTROL1 record in the Control file has an incorrect format.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** Contact your CICS system support person.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

| **CIU7050E** CINC Unable to retrieve information about journal *journal*.

| **Explanation:** An error has occurred during opening the *journal* journal.

| **System action:** The XML response containing the message is sent to the user.

| **User response:** Refer to the CIU2201S message.

| **Module:** CIUATM03

| **Destination**

| Client and CINT TD Queue.

CIU8000E The file you have selected could not be found. Please select another file.

Explanation: The file you chose to open cannot be found.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8001E File read error: an internal key value is not a valid integer. Please select another file.

Explanation: There was an error in reading the primary key from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8002E File read error: an internal key value is not a valid integer. Please select another file.

Explanation: There was an error in reading the secondary key from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8003E File read error: an internal key value is not a valid integer. Please select another file.

Explanation: There was an error in reading key3 from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8004E File read error: an internal key value is not a valid key value. Please select another file.

Explanation: There was an error in reading key4 from the specified file.

System action: Returns to wizard; no file is opened..

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8005E File read error: an internal key value is not a valid key value. Please select another file.

Explanation: There was an error in reading key2 from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8006E File read error: an internal key value is not a valid key value. Please select another file.

Explanation: There was an error in reading key3 from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8007E File read error: APPS must be a 3 character code.

Explanation: There was an error in reading the application code (APPS) from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8008E File read error: error reading the selected file.

Explanation: There was an error in reading SELECT (CSEL), WHERE (CWHE), or EQUALS (CEQU) values from the specified file.

System action: Returns to wizard; no file is opened.

User response: Open a different file.

Module: QueryMenu1.java

Destination

Client.

CIU8009E Error closing file.

Explanation: There was an error in closing an open file.

System action: Prints a message to the log; does not notify user.

User response: None.

Module: QueryMenu1.java

Destination

Client.

CIU8040E The URL provided in the CICS IA Preferences appears invalid. Please check the Host URL given in the CICS IA Preferences.

Explanation: While running a query, the Universal Resource Locator (URL) of the CICS TS region, specified in the client's CICS IA Preferences, was found to be invalid.

System action: The query cannot be executed. The query wizard remains open.

User response: Enter, in the client's CICS IA Preferences, the correct target address for the CICS TS region.

Module: QueryWizard.java

Destination

Client.

CIU8045E Communication with the host failed.

Explanation: A remote method call to CICS TS failed.

System action: The query cannot be executed. The query wizard remains open.

User response: Ensure that the CICS TS region is available, and that you have specified, in the client's CICS IA Preferences, the correct target address for the CICS TS region.

Module: QueryWizard.java

Destination

Client.

CIU8050E Communication with the host failed.

Explanation: CICS IA client could not communicate with the CICS TS region.

System action: The query cannot be executed. The query wizard remains open.

User response: Ensure that the CICS TS region is available, and that you have specified, in the client's CICS IA Preferences, the correct target address for the CICS TS region.

Module: QueryWizard.java

Destination

Client.

CIU8055E The data could not be displayed in a table. Please try the client again. If the problem persists contact IBM support.

Explanation: An error occurred while attempting to display, in a table, the data returned from a query on CICS resources.

System action: The query results cannot be displayed.

User response: Try the wizard again. If the error persists, contact your CICS support person.

Module: QueryEditorData.java

Destination

Client.

CIU8060E Copy fail: Cannot copy row numbers. Select another cell and try again.

Explanation: An attempt was made to copy a cell in the first column of the table, the row numbers. The copy is unsuccessful.

System action: The first column cannot be copied. No new data is copied to the system clipboard.

User response: Make a new selection from the table and try copy again.

Module: CopyAction.java

Destination

Client.

CIU8061E Copy fail: No Cell in the table was selected. Select an individual cell and try again.

Explanation: An attempt was made to copy an individual cell value, but no cell was selected in the table.

System action: None. No new data is copied to the system clipboard.

User response: Select an individual cell and try again.

Module: CopyAction.java

Destination

Client.

CIU8067E FileIO error: error saving *FILENAME*. Please try again.

Explanation: An error occurred creating a file or writing data to a file.

System action: A file can be created and no data written to the file, or there might be no system action.

User response: Try saving the file again, with a different file name. If the error persists, contact your CICS support representative.

Module: SaveQueryTableDataAction.java

Destination

Client.

CIU8068E FileIO error: error overwriting file. Please try again.

Explanation: An error occurred overwriting a saved query file.

System action: None. File will not be overwritten.

User response: Try saving the file again, with a different file name. If the error persists, contact your CICS support representative.

Module: SaveQueryTableDataAction.java

Destination

Client.

CIU8100I All available data for this query retrieved from the host.

Explanation: The query is successful. All data for this query is returned from the host.

System action: No message is displayed to the user. All data retrieved from the host is shown in a query table. The 'fetch next 4000 rows' button is disabled to indicate there are no more rows available.

User response: None.

Module: Rcode_Messages.java

Destination

Client.

CIU8101I There are more records to be retrieved.

Explanation: The query is successful. The first 4000 rows of data are returned from the host.

System action: No message is displayed to the user. The first 4000 rows of data returned from the host are shown in a query table. The 'fetch next 4000 rows' button is enabled to indicate more rows are available.

User response: Click the 'fetch next 4000 rows' button in the top right corner of the query editor to fetch up to the next 4000 rows.

Module: Rcode_Messages.java

Destination

Client.

CIU8102I No data to retrieve for this query.

Explanation: No data was found for this query.

System action: An information message is displayed to user. The query wizard remains open.

User response: None.

Module: Rcode_Messages.java

Destination

Client.

CIU8103W CICS region is not connected to DB2. See message CIU8200I in the CINT log.

Explanation: CICS IA has detected that there is no DB2 connection.

System action: A warning message is displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8104W Invalid Application code. See message CIU8202W in the CINT log.

Explanation: You have not entered a valid application code. The code must be 3 characters long and must match an application code defined to CICS IA.

System action: A warning message is displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8105W Input a valid 2 digit KEY1 field. See CIU8203W in the CINT log.

Explanation: You have entered an invalid code for KEY1 of the API call.

System action: A warning message is displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8106W Enter correct WHERE clause. See message CIU8204W.

Explanation: Your input includes one or more non-alphanumeric characters in a field that requires alphanumeric data.

System action: A warning message is displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Correct selection.

Module: Rcode_Messages.java

Destination

Client.

CIU8107W No columns selected for display. See message CIU8206W.

Explanation: You have not selected any columns to retrieve.

System action: A warning message is displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Select a column.

Module: Rcode_Messages.java

Destination

Client.

CIU8108E Bad CICS return code. See message CIU8205E in the CINT log.

Explanation: The CICS IA API received an invalid response when issuing an EXEC CICS command.

System action: An error message written to the error log and displayed to the user. The query wizard

remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8109E Bad SQL return code. See message CIU8201E in the CINT log.

Explanation: The CICS IA API has encountered an SQL error during processing.

System action: An error message written to the error log and displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8110E Client and server not compatible. See message CIU8207E in the CINT log.

Explanation: The client code and the server API CIUAQRYC are not at the same APAR level.

System action: An error message written to the error log and displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

CIU8111E No message file found on host.

Explanation: No server error message module could be found on the host. Ensure that the default language module CIUMSGE is defined and available to CICS.

System action: An error message written to the error log and displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Contact your CICS IA support representative.

Module: Rcode_Messages.java

Destination

Client.

**CIU8120E Invalid return code from the host.
Contact IBM support.**

Explanation: The return code from the host is not recognized within the CICS IA client.

System action: An error message written to the error log and displayed to the user. The query wizard remains open. No query data is returned from the host or displayed in a query table.

User response: Try to replicate this problem then contact your CICS support person.

Module: Rcode_Messages.java

Destination

Client.

CIU8200W CICS region is not connected to DB2

Explanation: CICS IA has detected that there is no DB2 connection.

System action: None.

User response: Contact your CICS system support person.

Module: CIUAQRYC

Destination

CINT log

**CIU8201E DB2 table error on function SQL code
code**

Explanation: SQL error *code* has occurred while executing DB2 function *function*.

System action: The transaction is terminated.

User response: Ensure that CICS IA is properly installed. Contact your system support group.

Module: CIUAQRYC

Destination

CINT log

CIU8202W Input a valid 3 digit application code.

Explanation: You have not entered a valid application code. The code must be 3-characters long and must match an application code defined to CICS IA.

System action: None.

User response: Enter a valid application code.

Module: CIUAQRYC

Destination

CINT log

CIU8203W Input a valid 2 digit KEY1 field.

Explanation: You have not entered a valid code for KEY1.

System action: None.

User response: Enter a valid code for KEY1. The valid codes are:

- 01 CICS
- 02 DB2
- 03 MQ
- 04 IMS
- 05 Affinities

Module: CIUAQRYC

Destination

CINT log

**CIU8204W One or more WHERE fields are not
alphanumeric.**

Explanation: Your input includes one or more non-alphanumeric characters in a field that requires alphanumeric data.

System action: None.

User response: Check your input, correct, and try again.

Module: CIUAQRYC

Destination

CINT log

**CIU8205S CICS command failed RESP=resp
RESP2=resp2**

Explanation: The CICS IA API received an invalid response when issuing the EXEC CICS *command* command.

System action: None.

User response: For further details of the exception *resp* refer to the *command* in the *CICS System Programming Reference* manual. For further information on how to solve system problems, refer to the *CICS Problem Determination Guide*.

Module: CIUAQRYC

Destination

CINT log

CIU8206W No columns SELECTED for display.

Explanation: You have not selected any columns to retrieve.

System action: None.

User response: You must select at least one column for the query to retrieve.

Module: CIUAQRYC

Destination

CINT log

CIU8207E Client and server not at the same service level.

Explanation: The CICS IA client and the CICS IA server API program , CIUAQRYC , are at different service levels.

System action: None.

User response: Ensure that the client is at the latest APAR level. You can review the APAR level of the client by:

From the Eclipse dialog select HELP.
 Select 'About Eclipse'
 Select 'plug-in details'
 Scroll down and select 'CICS IA Client'
 Select 'More Info'

Module: CIUAQRYC

Destination

CINT log

IUXA

Explanation: An unexpected error occurred when calling Collector program CIUTABM, from one of the Collector exit programs. Transaction CINB issues this abend on behalf of the exit program.

System action: The Collector is stopped.

User response: Refer to message CIU4100S.

Module: CIUCINB1

IUXB

Explanation: The data space has filled up. If the situation was detected by a Collector exit program, transaction CINB issues this abend on its behalf.

System action: The Collector is stopped.

User response: Refer to message CIU4200S.

Module: CIUCINT3, CIUCINT6, CIUCINB1

IUXD

Explanation: A USER type record for a specified user is not found in the CIUCNTL control file.

System action: The transaction is terminated.

User response: Create the USER record again. If the problem recurs, contact the IBM support.

Module: Refer to message CIU2277S

Destination

IUXE

Explanation: A USER type record for a specified user in the CIUCNTL control file has an incorrect format.

System action: The transaction is terminated.

User response: Contact your CICS system support person.

Module: CIUCINB1

Destination

Refer to message CIU2278S.

IUXF

Explanation: The CINC transaction was initiated in a way, which is not allowed. The CINC transaction can be initiated only from the 3270 terminal.

System action: The transaction is terminated.

User response: Use the proper methods to initiate CINC.

Module: Refer to message CIU2285E.

Destination

IUXG

Explanation: The CINC transaction was initiated on a version, release or modification of CICS, which is not supported by the Command Flow collector.

System action: The transaction is terminated.

User response: The Command Flow collector cannot be run on this CICS release.

Module: Refer to message CIU2287E

Destination

IUXI

Explanation: An abend occurred on a remote region.

System action: The Collector is not started on the remote region. The CINC transaction is terminated on a local region .

User response: Check CINT TD queue on the remote

| region to fix the problem. Start the CINC transaction again.

| **Module:** Refer to message CIU2329S.

| Destination

| IUXT

| **Explanation:** An unexpected error occurred when a program from the CINC transaction issued an EXEC CICS command related to the CINC Collector user exit. The command is one of ENABLE, DISABLE or EXTRACT EXIT.

| **System action:** The CINC Collector is not stopped.

| **User response:** Refer to message CIU2291S.

| **Module:** CIUACM61

| IUXU

| **Explanation:** The method of initiating the CINC transaction is incorrect.

| **System action:** The Command flow collector is not stopped.

| **User response:** Refer to message CIU2285E.

| **Module:** CIUACM10

| IUXV

| **Explanation:** A REGION type record is not found in the CIUCNTL control file.

| **System action:** The CINC transaction is terminated.

| **User response:** Refer to message CIU2280S.

| **Module:** CIUACM10

| IUXW

| **Explanation:** A USER type record in the CIUCNTL control file has an incorrect format for the specified user.

| **System action:** The CINC Collector is not stopped.

| **User response:** Refer to message CIU2291S.

| **Module:** CIUACM61

| IUXX

| **Explanation:** The internal error appears in the CINC transaction.

| **System action:** The Command flow collector is stopped.

| **User response:** Refer to message CIU2288S.

| **Module:** CIUACM10

| IUXY

| **Explanation:** A CONTROL1 type record is not found in the CIUCNTL control file.

| **System action:** The Command flow collector is stopped.

| **User response:** Refer to message CIU2279S.

| **Module:** CIUACM10

| IUXZ

| **Explanation:** A USER type record for the specified user is not found in the CIUCNTL control file.

| **System action:** The CINC transaction is not terminated.

| **User response:** Refer to message CIU2277S.

| **Module:** CIUACM10

IUYA

Explanation: Transaction CINB received an unrecognisable request from another Collector component, CINT or a Collector exit program.

System action: The Collector is stopped.

User response: Refer to message CIU3302S.

Module: CIUCINB1

IUYC

Explanation: Transaction CINB received a request from another Collector component, CINT or an exit program, to abend because of an unexpected error.

System action: The Collector is stopped.

User response: Refer to message CIU3304S.

Module: CIUCINB1

IUYE

Explanation: A Collector program found an invalid dependency file number in an internal array in the Collector GWA.

System action: The Collector is stopped.

User response: Refer to message CIU3310S.

Module: CIUCINB2, CIUCINT3

IUYG

Explanation: Transaction CINB was still running at CICS termination.

System action: The Collector is stopped.

User response: Refer to message CIU3312S.

IUYH • IUZ8

Module: CIUCINB1

IUYH

Explanation: A Collector program found that the address held in the Collector GWA for one of the Collector internal modules was invalid.

System action: The Collector is stopped.

User response: Refer to message CIU3313S.

Module: CIUCINT4, CIUCINT5, CIUCINB1

IUYI

Explanation: An unexpected error occurred when calling Collector program CIUTABM to access dependency table data in the data space, from transaction CINT or CINB.

System action: The Collector is stopped.

User response: Refer to message CIU3314S.

Module: CIUCINB2, CIUCINT6

IUYJ

Explanation: The dependency data file has filled up.

System action: The Collector is stopped.

User response: Refer to message CIU3315S.

Module: CIUCINB2

IUYK

Explanation: Transaction CINB received an unrecognized resource type from another Collector component.

System action: The Collector is stopped.

User response: Refer to message CIU3316E.

Module: CIUCIND

IUZ1

Explanation: When the Collector was being started by transaction CINT, the header record could not be found on the VSAM dependency data file.

System action: The Collector is stopped.

User response: Refer to message CIU2230S.

Module: CIUCINT3

IUZ3

Explanation: Transaction CINT or CINB is running on a release of CICS which does not support the Collector.

System action: The Collector is stopped.

User response: Refer to message CIU2232E.

Module: CIUCINT1, CIUCINB1

IUZ4

Explanation: Records in control file CIUCNTL have incorrect format.

| **System action:** Transaction CINT or CINC is terminated.

User response: Refer to message CIU2209S.

Module: Refer to message CIU2209S.

IUZ5

Explanation: When transaction CINT attempted to start the Collector it found that a program or transaction exclude list had invalid contents.

System action: Transaction CINT is terminated.

User response: Refer to message CIU2244S.

Module: CIUA110C

IUZ6

| **Explanation:** Transaction CINT or CINC detected an unexpected error from a CICS dump domain function.

| **System action:** Transaction CINT or CINC is terminated.

User response: Refer to message CIU2245S.

| **Module:** Refer to message CIU2245S.

Destination

CINT TD queue.

IUZ7

Explanation: An unexpected error was detected when the CICS IA interface to Natural interacted with Natural SYSRDC.

System action: The CICS IA interface to Natural is stopped.

User response: Refer to message CIU4307S.

Module: CIURDCX1

IUZ8

Explanation: An unexpected error occurred when the CICS IA interface to Natural issued an EXEC CICS command.

System action: The CICS IA interface to Natural is stopped.

User response: Refer to message CIU4308S.

Module: CIURDCX1

IUZ9

Explanation: An abend occurred within the CICS IA interface to Natural.

System action: The CICS IA interface to Natural is stopped.

User response: Refer to message CIU4309S.

Module: CIURDCX1

IUZA

Explanation: An unexpected error occurred when issuing an EXEC CICS command, by a program from transaction CINT, CINB or CINC.

System action: The Collector is stopped.

User response: Refer to message CIU2201S.

Module: Refer to message CIU2201S.

IUZB

Explanation: An unexpected error occurred when issuing a VSAM file control EXEC CICS command, by a program from transaction CINT, CINB or CINC.

System action: The Collector is stopped.

User response: Refer to message CIU2202S.

Module: Refer to message CIU2202S.

IUZC

Explanation: The internal field holding the Collector state has an invalid value.

System action: The Collector is stopped.

User response: Refer to message CIU2203S.

Module: CIUCINT1, CIUCINT2

IUZD

Explanation: One of the files contains a CICS APPLID that does not match the APPLID of the CICS system.

System action: The Collector is stopped.

User response: Refer to message CIU2205S.

Module: CIUCINT1, CIUCINT2

IUZF

Explanation: An unexpected error error occurred when issuing a Collector user exit related EXEC CICS command, by a program from transaction CINT or CINB. The command is one of ENABLE, DISABLE or EXTRACT EXIT.

System action: The Collector is stopped.

User response: Refer to message CIU2206S.

Module: CIUCINT1, CIUCINT2, CIUCINT3, CIUCINT4, CIUCINT5, CIUCINT6, CIUCINB1

IUZH

Explanation: An unexpected error occurred when calling Collector program CIUTABM to create the MVS data space to hold the dependency data, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2210S.

Module: CIUCINT3

IUZI

Explanation: An unexpected error occurred when calling Collector program CIUTABM to create a dependency table in the data space, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2211S.

Module: CIUCINT3, CIUCINT6

IUZJ

Explanation: An unexpected error occurred when calling Collector program CIUTABM to add an element to a dependency table in the data space, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2212S.

Module: CIUCINT3

IUZN

Explanation: An unexpected error occurred when calling Collector program CIUTABM to destroy the data space, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2216S.

Module: CIUCINT4

IUZO

Explanation: An unexpected error occurred when calling Collector program CIUTABM to destroy a table in the data space, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2217S.

Module: CIUCINT6

IUZQ

Explanation: An unexpected error occurred when calling Collector program CIUCINP to create its MVS CPOOL storage, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2220S.

Module: CIUCINT3

IUZR

Explanation: An unexpected error occurred when calling Collector program CIUCINP to access its MVS CPOOL storage, from transaction CINT or CINB.

System action: The Collector is stopped.

User response: Refer to message CIU2221S.

Module: CIUCINT4, CIUCINT5, CIUCINB1

IUZS

Explanation: An unexpected error occurred when calling Collector program CIUCINP to destroy its MVS CPOOL storage, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2222S.

Module: CIUCINT4

IUZU

Explanation: An unexpected error occurred when calculating what percentage of the data space is occupied by dependency data, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2224S.

Module: CIUCINT1

IUZV

Explanation: The method of initiating transaction CINT is incorrect.

System action: The Collector is stopped.

User response: Refer to message CIU2225E.

Module: CIUCINT1

IUZY

Explanation: An unexpected error occurred when calling Collector program CIUTABM to replace a table element in the data space, from transaction CINT or CINB.

System action: The Collector is stopped.

User response: Refer to message CIU2228S.

Module: CIUCINT3, CIUCINB2

Collector table manager diagnostics

This section lists the meaning for each possible value of the call parameters that are included in the error messages issued if an error occurs on a call to the Collector table manager, CIUTABM.

Function code values

A list of function code values.

AUTM_CREATE_POOL	1
AUTM_DESTROY_POOL	2
AUTM_CREATE_TABLE	3
AUTM_DESTROY_TABLE	4
AUTM_ADD_ELEMENT	5
AUTM_DELETE_ELEMENT	6
AUTM_REPLACE_ELEMENT	7
AUTM_GET_KEY_ELEMENT	8
AUTM_GET_FIRST_ELEMENT	9
AUTM_GET_NEXT_ELEMENT	10
AUTM_GET_ELEMENT	11
AUTM_GET_KEY_GE_ELEMENT	12

Table identifier values

A list of table identifier values.

AUTM_CICS	1
AUTM_DB2	2
AUTM_MQ	3
AUTM_IMS	4
AUTM_DTP	5

AUTM_MQX	6
AUTM_CICL	7
AUTM_IMSX	8
AUTM_RESOURCE	9
AUTM_EDSR	11
AUTM_EDST	12
AUTM_EDR	13
AUTM_EDT	14
AUTM_TSQ	15
AUTM_TST	16
AUTM_LRP	17
AUTM_LRT	18
AUTM_SRS	19
AUTM_SRT	20
AUTM_CWA	21
AUTM_CWT	22
AUTM_GFA	23
AUTM_GFM	24
AUTM_LFA	25
AUTM_LFM	26
AUTM_ICR	27
AUTM_ICM	28
AUTM_SPI	29
AUTM_WAIT	30
AUTM_TT	31
AUTM_UT	32
AUTM_PT	33
AUTM_AT	34
AUTM_LT	35
AUTM_BLD_DNT	38
AUTM_BLD_GNT	39
AUTM_BLD_TT	40
AUTM_BLD_MERGED	41

Reason code values

A list of reason code values.

AUTM_INVALID_FUNCTION	0
AUTM_NO_STORAGE	1
AUTM_ELEMENT_NOT_FOUND	2
AUTM_ELEMENT_EXISTS	3
AUTM_INVALID_TABLE	4
AUTM_IEFUSI_HIT	5
AUTM_TABLE_EXISTS	6
AUTM_TABLE_DOES_NOT_EXIST	7
AUTM_POOL_EXISTS	8
AUTM_POOL_DOES_NOT_EXIST	9
AUTM_INVALID_CURSOR	10
AUTM_DEFAULT_SIFD_ERROR	192
AUTM_DEFAULT_SIFA_ERROR	193
AUTM_DEFAULT_DSP_ERROR	194
AUTM_DEFAULT_AVL_ERROR	195
AUTM_SIFD_CREATE_POOL_ERROR	196
AUTM_SIFA_CREATE_POOL_ERROR	197
AUTM_DSP_CREATE_POOL_ERROR	198
AUTM_SIFD_DESTROY_POOL_ERROR	199
AUTM_SIFA_DESTROY_POOL_ERROR	200
AUTM_DSP_DESTROY_POOL_ERROR	201
AUTM_AVL_CREATE_TABLE_ERROR	202
AUTM_SIFA_CREATE_TABLE_ERROR	203
AUTM_AVL_DESTROY_TABLE_ERROR	204
AUTM_SIFA_DESTROY_TABLE_ERROR	205
AUTM_AVL_ADD_ERROR	206
AUTM_SIFA_ADD_ERROR	207
AUTM_AVL_GET_KEY_ERROR	208
AUTM_AVL_GET_FIRST_ERROR	209

AUTM_AVL_GET_NEXT_ERROR	210
AUTM_AVL_DELETE_ERROR	211
AUTM_SIFA_DELETE_ERROR	212
AUTM_AVL_REPLACE_ERROR	213
AUTM_DSP_RESERVE_ERROR	214
AUTM_DSP_RELEASE_ERROR	215
AUTM_DSPSERV_CREATE_ERROR	216
AUTM_DSPSERV_DELETE_ERROR	217
AUTM_ALESERV_ADD_ERROR	218
AUTM_ALESERV_DELETE_ERROR	219
AUTM_AVL_GET_ERROR	220

Collector CINB request queue manager diagnostics

This section lists the meaning for each possible value of the call parameters that are included in the error messages issued if an error occurs on a call to the Collector CINB request queue manager, CIUCINP.

Function code values

A list of function code values.

AUCP_ADD_CELL_FIRST	1
AUCP_ADD_CELL_LAST	2
AUCP_CREATE_CPOOL	3
AUCP_DESTROY_CPOOL	4
AUCP_GET_CELL	5

Reason code values

A list of reason code values.

AUCP_NO_STORAGE	1
AUCP_CPOOL_FAILED	2
AUCP_INVALID_FUNCTION	3
AUCP_NOT_FOUND	4

Date formatter diagnostics

This section lists the meaning for each possible value of the call parameters that are included in the error messages issued if an error occurs on a call to the Interdependency Analyzer date formatter, CIUCINDT.

Reason code values

CIUD_NO_DATE	1
CIUD_FORMAT	2

Appendix E. Worksheet for the installation customization program

This section contains a worksheet for use with the installation customization program.

The installation customization program allows you to create customized installation jobs in which the names of system entities, such as the high-level qualifier (hlq) of the CICS IA data sets, are set to specified values to suit your local environment. The worksheet consists of a table of installation variables, such as `_dbid_`, the identifier of the DB2 database, that can be passed to the installation customization program. You can record the value that you assign to each variable in the **Value** column of the table.

Note: If you are creating the IA DB2 database on V8.1 of DB2 in compatibility mode, enter V710 into the DB2 version variable in the configuration exec.

The installation customization program is described in “Running the installation customization program” on page 44.

Table 111. Variables that can be passed to the installation customization program

Section	Variable	Description	Value
Output datasets	_outsmp1_ REQUIRED	OUTPUT DSN FOR SCIUSMP1 The output data set that contains the modified hlq.SCIUSAMP members except the database related ones. Default value is SYSDA.	
	outsmp2 REQUIRED	OUTPUT DSN FOR SCIUSMP2(DB2) The output data set that contains the modified database related hlq.SCIUSAMP members.	
	outsamn REQUIRED	OUTPUT DSN FOR SCIUSAME/K The output data set that contains the modified hlq.SCIUSAME/K members.	
	outclis REQUIRED	OUTPUT DSN FOR SCIUCLIS The output data set that contains the modified hlq.SCIUCLIS members.	
	outsq1 REQUIRED	OUTPUT DSN FOR SCIUSQL The output data set that contains the modified hlq.SCIUSQL members.	
	outdat1 REQUIRED	OUTPUT DSN FOR SCIUDAT1 The output data set that contains the copied hlq.SCIUDAT1 members.	
	outdat2 REQUIRED	OUTPUT DSN FOR SCIUDAT2 The output data set that contains the copied hlq.SCIUDAT2 members.	
	vvv REQUIRED	OUTPUT DEVICE TYPE The output device on which the CICS IA VSAM files are catalogued.	

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
CICS IA variables	_hlq_ OPTIONAL	IA PRODUCT QUALIFIER The high-level qualifier (hlq) of the CICS IA data sets. Passed as input to the CIUCNFG1 program.	
	ciuhlq REQUIRED	IA VSAM FILE QUALIFIER The data set qualifier for the CICS IA VSAM and output files.	
	ciuhlqc OPTIONAL	IA CSV FILE QUALIFIER The data set qualifier for QSAM CSV files.	
	lsnamhlq REQUIRED	IA LOG STREAM QUALIFIER	
	lscfdasd REQUIRED	COUPLING FACILITY or DASD Specify CF if the logstream to be managed by Coupling Facility, or DASD for a DASD-only logstream definition. Default value is CF.	
	lshlq REQUIRED	LOG STREAM DATASET NAME HLQ The high-level qualifier for both the log stream data set name and the staging data set name. Default value is IXGLOGR.	
	lsstrcnm OPTIONAL	IA LOG STREAM STRUCTNAME The name of the coupling facility structure associated with the coupling facility log stream being defined. Default value is LOG_CIUMTJNL_001.	
	lsretpd OPTIONAL	LOG STREAM RETENTION PERIOD The retention period, in days, for log data in the log stream. Default value is 24.	

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
CICS IA variables (continued)	_ciugd_ REQUIRED	IA GDG DATASET QUALIFIER The high-level qualifier for the GDG data set name.	
	smsdatac OPTIONAL	IA VSAM FILE DATACLASS The DFSMS data class to be used for CICS IA file definitions.	
	smsstorc OPTIONAL	IA VSAM FILE STORAGECLS The DFSMS storage class to be used for CICS IA file definitions.	
	smsgc OPTIONAL	IA VSAM FILE MNGMNTCLASS The DFSMS management class to be used for CICS IA file definitions.	
	ciuspace REQUIRED	IA VSAM FILE SPACE UNITS The space units that reflect the data set size.	
	ciupquan REQUIRED	IA VSAM FILE PRIMARY QTY	
	ciusquan OPTIONAL	IA VSAM FILE SECNDRY QTY	

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
DB2 variables	_db2hlq_ REQUIRED	DB2 LOAD DATASET NAME The name of the DB2 data set for SDSNLOAD.	
	db2run REQUIRED	DB2 RUNLIB DATASET NAME The name of the DB2 data set for RUNLIB.LOAD.	
	db2hlq2 REQUIRED	DB2 PROCLIB DATASET NAME The data set of the DB2 DB2PROC library.	
	db2vcat REQUIRED	DB2 VCAT QUALIFIER The qualifier of the CICS IA DB2 data sets.	
	dbid REQUIRED	DB2 SUB SYSTEM The identifier of the DB2 database.	
	qual REQUIRED	DB2 TABLE QUALIFIER The implicit qualifier to be used with unqualified names of DB2 tables, views, and so on.	
	own REQUIRED	DB2 TABLE OWNER The ID of the owner of the DB2 plans for authorization purposes.	
	vol REQUIRED	DB2 DASD VOLUME The DASD volume on which DB2 tables are to be created: for example, "SYSDA".	
	vv REQUIRED	DB2 VERSION Identifies the DB2 plan DSNTEP_vvvv_. Four characters representing the DB2 version number; for example, V810. Note: If you are running in DB2 Version 8 compatibility mode select V710.	
racgrp OPTIONAL	DB2 RACF GROUP The RACF group that is authorized to access the CICS IA DB2 plans.		

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
DB2 variables (continued)	_db2bfpt_ REQUIRED	DB2 BUFFERPOOL FOR TBSPC The BUFFERPOOL value, used in CREATE TABLESPACE statements. Default value is BP0.	
	db2bfpi REQUIRED	DB2 BUFFERPOOL FOR INDEX The BUFFERPOOL value, used in CREATE INDEX statements. Default value is BP0.	
	db2stgt REQUIRED	DB2 STOGROUP FOR TBLSPCS The STOGROUP value, used in CREATE TABLESPACE statements.	
	db2stgi REQUIRED	DB2 STOGROUP FOR INDEXES The STOGROUP value, used in CREATE INDEX statements.	
	db2styn REQUIRED	USE EXISTING STORAGE GROUP Specify YES to keep the existing storage groups or NO to generate a creation of new ones.	
	db2dbnt REQUIRED	DB2 DATABASE FOR TBLSPCS The database name to host all tablespaces.	
	db2planc OPTIONAL	DB2 PLAN NAME FOR CICS The plan name for CICS programs. Default value is CIUCICS.	
	db2planb OPTIONAL	DB2 PLAN NAME FOR BATCH The plan name for batch programs. Default value is CIUBTCH.	
	ccsid REQUIRED	DB2 CCSID (CODE PAGE) Code character set identifiers (CCSIDs) for the EBCDIC input file of LOAD utility. Default value is 00000.	
DB2 variables (continued)	_wlm_ REQUIRED	DB2 WLM PROCEDURE NAME WLM (workload manager) environment, in which the stored procedure is to run.	
	db2pack REQUIRED	DB2 COLLECTION ID The name of the collection of the package the CICS IA programs to be included into. Default value is CPS4.	

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
CICS variables	_cicsver_ OPTIONAL	CICS VERSION The product version of this CICS Transaction Server region: V130, V220, V230, V310, V320, V410 or V420. Default value is V410.	
	cicshlq REQUIRED	CICS QUALIFIER The data set qualifier of the CICS TS SDFHLOAD library.	
	csdname REQUIRED	CICS CSD FILE NAME The name of the CICS System Definition (CSD) data set.	
	list REQUIRED	CICS CSD LIST NAME The CSD group list to which to append the CICS IA resource group.	
	ciugrpt REQUIRED	CICS CSD COLLECTOR GROUP The group name for CINT resource definitions.	
	ciugrpf REQUIRED	CICS LOCAL FILES GROUP The group name for file resource definitions defined locally.	
	ciugrpr REQUIRED	CICS REMOTE FILES GROUP The group name for file resource definitions defined remotely.	
	groupw REQUIRED	CICS WEB SUPPORT GROUP The group name for web resource definitions.	
	rls REQUIRED	RLSACCESS FOR VSAM FILES A variable that defines whether RLS is used with VSAM files.	
	lsr REQUIRED	LSRPOOLID FOR VSAM FILES The identifier of the LSRPOOL for the VSAM file definitions.	
CICS variables (continued)	_for_ OPTIONAL	CICS FILE OWNING REGION If you use a CICS file-owning region (FOR) to share the CICS IA VSAM files, the system identifier (SYSID) of the FOR.	
	libnm REQUIRED	LIBRARY RESOURCE NAME (for SCIULOAD) CICS LIBRARY resource name for defining SCIULOAD dataset (used for CICS TS version 3.2 or higher).	
	libnmek REQUIRED	LIBRARY RESOURCE NAME (for SCIULODE/K) CICS LIBRARY resource name for defining SCIULODE/K dataset (used for CICS TS version 3.2 or higher).	

Table 111. Variables that can be passed to the installation customization program (continued)

Section	Variable	Description	Value
Migration variables	OPTIONAL	PREVIOUS RELEASE The previous release of CICS IA. Values can be 3.1.	
	OPTIONAL _ciuhlqo_	PREVIOUS IA QUALIFIER The high-level data set qualifier of the VSAM and output files used in the previous version of CICS IA.	
	OPTIONAL _ciuhlqn_	NEW IA QUALIFIER The high-level data set qualifier of the VSAM and output files to be used in the new version of CICS IA.	
	OPTIONAL _oldqual_	OLD DB2 TABLE QUALIFIER The implicit qualifier used with unqualified names of DB2 tables, views, and other names, in the previous version of IA.	
	OPTIONAL _newqual_	NEW DB2 TABLE QUALIFIER The implicit qualifier to be used with unqualified names of DB2 tables, views, and other names, in the new version of IA.	
	REQUIRED _tcpport_	TCP IP PORT The TCP/IP port used by Atom service to control the collector through CICS Explorer.	
	REQUIRED _atomcfg1_	ATOM CONFIG FILE LOCATION The location of the Atom service configuration file in the HFS file system. This is the first part of the HFS directory path that contains the CIUATM01 configuration file. The default will be " <i>pathprefix</i> /usr/lpp/cicsia/ <i>cuiver</i> ". The <i>pathprefix</i> and <i>cuiver</i> are variables used during the SMP/E install of CICS IA. This value will be concatenated with IBM/atomservices/config/CIUATM01 to define the location of the Atom service Configuration file for CICS IA.	
General variables	OPTIONAL _asmlr_	ASSEMBLER PROGRAM The assembler program: ASMA90 (default) or IEV90.	
	REQUIRED _lehlq_	LE QUALIFIER The high-level data set qualifier of the SCEERUN library.	

Table 112 on page 325 lists variables that cannot be customized by the installation customization program.

Table 112. Names that are not customized by the installation customization program

	Variable	Description	Value
Names that are not customized by the installation customization program	_xxx_	CICS IA application suffixes.	
	cauhlq	The high-level data set qualifier of the VSAM affinities file.	
	applid	The CICS region APPLID used in reporting and load jobs.	
	ciudet	The data set qualifier of the output data set used by the CICS IA Load Module Scanner.	
	scan	The data set name of the load module to be scanned by the CICS IA Load Module Scanner.	

Appendix F. CICS IA space considerations

This appendix contains information on how to calculate the space requirements for the following:

- “Data space allocation” on page 328
- “VSAM data set allocation” on page 331
- “DB2 space allocation” on page 333

Required data

To calculate the space allocation requirements, estimate some values for the environment in which you want to run CICS IA.

Refer to the worksheet supplied in Table 113 and Table 114 on page 328. These values are used in later calculations.

Table 113. Values required for each CICS region

Required Data	Description	Value
PROG_TRAN_RATIO	Estimate of the number of transactions and program associations. For example, program PROGA can be associated with more than one transaction.	
NUM_CICS_PROG	Number of programs that contain EXEC CICS commands.	
AVG_EXEC_LONG	Average number of EXEC CICS calls, per program, that have a data space key greater than 32. These commands are the EXEC CICS ENQ and EXEC CICS DEQ.	
AVG_EXEC_SHORT	Average number of EXEC CICS calls, per program, that have a data space key less than or equal to 32. These commands are all EXEC CICS commands apart from EXEC CICS ENQ and EXEC CICS DEQ commands above.	
AVG_DTP	Average number of EXEC CICS DTP calls, ALLOCATE, CONNECT, SEND, CONVERSE, and FREE commands.	
NUM_DB2_PROG	Number of programs that contain EXEC SQL commands.	
AVG_DB2	Average number of EXEC SQL calls per program.	
NUM_IMS_PROG	Number of programs that contain EXEC IMS commands.	
AVG_IMS	Average number of EXEC IMS calls per program.	
NUM_MQ_PROG	Number of programs that contain EXEC MQ commands.	
AVG_MQ	Average number of EXEC MQ calls per program.	
NUM_AFF_PROG	Number of programs that contain EXEC AFF commands.	

Table 113. Values required for each CICS region (continued)

Required Data	Description	Value
AVG_AFF	Average number of EXEC AFF calls per program.	
NUM_NATURAL_PROG	Number of Natural programs that contain ADABAS or PROGRAM calls.	
AVG_NATURAL	Average number of ADABAS or PROGRAM calls per Natural program.	

Table 114. Values required for VSAM and DB2 calculations

Required Data	Description	Value
NUM_REGION_V	Number of CICS regions sharing the VSAM file. This value is required to calculate the VSAM file space allocation.	
NUM_REGION_D	Total number of CICS regions in which CICS IA is collecting. This value is required to calculate the DB2 table and index sizes. The CICS IA DB2 database can contain data from more than one set of VSAM files.	

Data space allocation

The data space consists of a number of tables, both permanent and temporary. The number and size of the tables varies depending on whether you are collecting Interdependency data or Affinities data.

About this task

The size of the data space used by CICS IA is defined in the 'General Options' panel.

1. Select option 2 from the CINT transaction main menu.
2. Select option 6 against the region or against the default record. The size can vary from 10 MB to 2000 MB; the default is set to 16 MB.

To view the percentage of the data space used in the statistics panel for each region

1. Select option 1 in the CINT transaction main menu.
2. Select option 5 against the region. The data space name is 00000INT.

Calculating the space required for interdependency collection

The interdependency collection can consist of up to eight tables depending on the options selected.

This section describes how to calculate the space required for Interdependency Collection.

Calculating CICS data space

Three tables are used to create CICS data.

About this task

Collect CICS data as described below:

1. Estimate the number of application programs that are used in the CICS region, using the CICS IA scanner. Call this number NUM_CICS_PROG.
2. Estimate the average number of EXEC CICS calls per program, using the CICS IA scanner. Run the scanner against all load module data sets in the DFHRPL that make up your applications. There are three types:
 - a. Key less than or equal to 32 bytes. Including all EXEC CICS calls other than those listed in step b. below. Call this number AVG_EXEC_SHORT.
 - b. Resources greater than 32 characters. Currently only EXEC CICS ENQ or EXEC CICS DEQ have resource names greater than 32 bytes. Call this number AVG_EXEC_LONG.
 - c. Resources that use DTP. These resources consist of the following commands: ALLOCATE, CONNECT, SEND, CONVERSE and FREE. Call this number AVG_DTP.
3. Estimate the number of transaction and program associations. In this example, the estimate is that each can be associated with five transactions. Call this number PROG_TRAN_RATIO.
4. Calculate CICS data space as follows:
$$(108 * \text{NUM_CICS_PROG} * \text{AVG_EXEC_SHORT} * \text{PROG_TRAN_RATIO}) + (338 * \text{NUM_CICS_PROG} * \text{AVG_EXEC_LONG} * \text{PROG_TRAN_RATIO}) + (16 * \text{NUM_CICS_PROG} * \text{AVG_DTP} * \text{PROG_TRAN_RATIO}) = \text{CICS_DS bytes.}$$

Calculating DB2 data space

One table is used to store DB2 data.

About this task

Collect DB2 data as follows:

1. Estimate the number of application programs that are used in the CICS region that use DB2. Call this number NUM_DB2_PROG.
2. Estimate the average number of EXEC SQL calls per program. Call this number AVG_DB2.
3. Estimate the number of transaction and program associations. In this example the estimate is that each can be associated with five transactions.
4. Calculate DB2 data space as follows:
$$(120 * \text{NUM_DB2_PROG} * \text{AVG_DB2} * \text{PROG_TRAN_RATIO}) = \text{DB2_DS bytes}$$

Calculating IMS data space

Two tables are used to store IMS data.

About this task

Collect IMS data as follows:

1. Estimate the number of application programs that are used in the CICS region that use DLI calls. Call this number NUM_IMS_PROG.
2. Estimate the average number of EXEC DLI calls per program. Call this number AVG_IMS.
3. Estimate the number of transaction and program associations. In this example, the estimate is that each can be associated with five transactions.
4. Calculate IMS data space as follows:

$$(75 * \text{NUM_IMS_PROG} * \text{AVG_IMS} * \text{PROG_TRAN_RATIO}) + \\ (20 * \text{NUM_IMS_PROG} * \text{AVG_IMS} * \text{PROG_TRAN_RATIO}) = \text{IMS_DS bytes.}$$

Calculating MQ data space

Two tables are used to store MQ data.

About this task

Collect MQ data as follows:

1. Estimate the number of application programs that are used in the CICS region that use MQ. Call this number NUM_MQ_PROG.
2. Estimate the average number of MQ calls per program. Call this number AVG_MQ.
3. Estimate the number of transaction and program associations. In this example, the estimate is that each can be associated with five transactions.
4. Calculate MQ data space as follows:

$$(103 * \text{NUM_MQ_PROG} * \text{AVG_MQ} * \text{PROG_TRAN_RATIO}) + \\ (60 * \text{NUM_MQ_PROG} * \text{AVG_MQ} * \text{PROG_TRAN_RATIO}) = \text{MQ_DS bytes}$$

Calculating Natural data space

One table is used to store Natural data.

About this task

Collect Natural data as follows:

1. Estimate the number of Natural programs that contain ADABAS or PROGRAM calls. Call this number NUM_NATURAL_PROG.
2. Estimate the average number of ADABAS or PROGRAM calls per Natural program. Call this number AVG_NATURAL.
3. Estimate the number of transaction and program associations. In this example, the estimate is that each can be associated with five transactions.
4. Calculate Natural data space as follows:

$$(144 * \text{NUM_NATURAL_PROG} * \text{AVG_NATURAL} * \text{PROG_TRAN_RATIO}) = \text{NATURAL_DS bytes}$$

Calculating the resource data space

There is one table used to capture the CICS resource data.

About this task

To collect the CICS resource data:

Procedure

1. Estimate the number of each of the following resources used in the CICS region:
 - File resources. Call this number NUM_CICS_FILE.
 - Program resources. Call this number NUM_CICS_PROG.
 - Transaction resources. Call this number NUM_CICS_TRAN.
 - Transient Data Queue resources. Call this number NUM_CICS_TDQ.
 - Temporary Storage Queue resources. Call this number NUM_CICS_TSQ.
 - Web services resources. Call this number NUM_CICS_WEBS
 - TRUE and GLUE exits. Call this number NUM_CICS_EXIT
2. Calculate the CICS data space:

```

(NUM_CICS_FILE +
 NUM_CICS_PROG +
 NUM_CICS_TRAN +
 NUM_CICS_TDQ +
 NUM_CICS_TSQ +
 NUM_CICS_WEBS +
 NUM_CICS_EXIT) * 165 = CICS_RES_DS bytes

```

Total Interdependency data space calculation for Interdependency

Calculation for the total data space required for interdependency.

$CICS_DS + DB2_DS + IMS_DS + MQ_DS + NATURAL_DS + CICS_RES_DS = TOTAL_DS$ in bytes

Calculating the space required for affinity collection

A number of data spaces are tables used when collecting affinity information.

About this task

Use the average length for the calculation, 200 bytes plus another 100 bytes for temporary tables. Using the average length removes the requirement to break the calculation down to each table. Collect affinity data as follows:

1. Estimate the number of programs that possibly have affinity commands using the IA scanner. Call this number NUM_AFF_PROG.
2. Estimate the number of affinity commands per program using the IA scanner. Call this number AVG_AFF.
3. Calculate for Affinity data space as follows:
 $(300 * NUM_AFF_PROG * AVG_AFF) = NUM_AFF_DS$ in bytes

VSAM data set allocation

There are ten VSAM files associated with CICS IA.

The JCL to create these files is in SCIUSMP1 members CIUJCLCC and CIUJCLCA.

The allocation parameters of the VSAM files can be customized by the installation customization program. The values for the following parameters can be specified:

- VSAM file data class
- VSAM file storage class
- VSAM file management class
- VSAM file space units (CYLINDERS, TRACKS, RECORDS, KILOBYTES or MEGABYTES)
- VSAM file primary quantity, in the space units specified
- VSAM file secondary quantity, in the space units specified

See Appendix E, "Worksheet for the installation customization program," on page 317 for a list of the variables for the above parameters.

Control file: CIUCNTL

Maximum record length is 732 bytes. Number of records is the number of CICS regions in the shared environment. Call this number NUM_REGION_V.

$Size = ((732 * (NUM_REGION_V+1)) + 28)$ bytes

Dependency files: CIUINT1, 2, 3, 4, 5, 6, 7

The size of the VSAM files can be calculated with the formulas listed.

The formulas refer to the information collected in "Data space allocation" on page 328.

CIUINT1 - CICS

$$\text{Size} = ((97 * \text{NUM_REGION_V}) + (127 * \text{NUM_CICS_PROG} * \text{AVG_EXEC_SHORT} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT2 - DB2

$$\text{Size} = ((110 * \text{NUM_REGION_V}) + (139 * \text{NUM_DB2_PROG} * \text{AVG_DB2} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT3 - MQ

$$\text{Size} = ((103 * \text{NUM_REGION_V}) + (132 * \text{NUM_MQ_PROG} * \text{AVG_MQ} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT4 - IMS

$$\text{Size} = ((65 * \text{NUM_REGION_V}) + (94 * \text{NUM_IMS_PROG} * \text{AVG_IMS} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT5 - CICS

$$\text{Size} = ((273 * \text{NUM_REGION_V}) + (357 * \text{NUM_CICS_PROG} * \text{AVG_EXEC_LONG} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT6 - CICS resources

$$\text{Size} = ((97 * \text{NUM_REGION_V}) + (298 * \text{NUM_CICS_FILE}) + (204 * \text{NUM_CICS_PROG}) + (203 * \text{NUM_CICS_TRAN}) + (249 * \text{NUM_CICS_TDQ}) + (148 * \text{NUM_CICS_TSQ}) + (1208 * \text{NUM_CICS_WEBS}) + (100 * \text{NUM_CICS_EXIT})) \text{ bytes}$$

CIUINT7 - Natural

$$\text{Size} = ((135 * \text{NUM_REGION_V}) + (159 * \text{NUM_NATURAL_PROG} * \text{AVG_NATURAL} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

Affinity files: CIUAFF1,2,3

The calculation considers that the average number of affinity commands per program is split into the three files, with the ratio 50%, 40% and 10%.

CIUAFF1 – Affinity (Key <= 16)

$$\text{Size} = ((47 * \text{NUM_REGION_V}) + (255 * \text{NUM_AFF_PROG} * (\text{AVG_AFF}/2) * \text{NUM_REGION_V})) \text{ bytes}$$

CIUAFF2 – Affinity (Key <= 32)

$$\text{Size} = ((63 * \text{NUM_REGION_V}) + (255 * \text{NUM_AFF_PROG} * (\text{AVG_AFF}/10*4) * \text{NUM_REGION_V})) \text{ bytes}$$

CIUAFF3 – Affinity (Key > 32)

Size = ((255 * NUM_REGION_V) +
(255 * NUM_AFF_PROG * (AVG_AFF/10) * NUM_REGION_V)) bytes

DB2 space allocation

Calculate the DB2 space allocations using these assumptions and guidance.

In the following calculations, the following assumptions apply:

- All data spaces and indexes in CICS IA use Bufferpool BP0, which is a 4 KB page.
- Number of bytes available in a 4 KB page is 4089.
- You have to calculate the PRIQTY for all tables and indexes.
- SECQTY is set to 10% of the PRIQTY. For more information on SECQTY, see the *DB2 Administration Guide*.

To calculate the PRIQTY for tablespaces and indexes, use the calculation in Figure 57:

```
ROWS_PER_PAGE = 4089/ROW_SIZE
FREE_BYTES = 4089/100 * PERCENT_FREE
FREE_ROWS = FREE_BYTES/ROW_SIZE

NUM_OF_4K_PAGES = NUM_ROWS / (ROWS_PER_PAGE - FREE_ROWS)
PRIQTY = 4 * NUM_OF_4K_PAGES
```

Figure 57. Calculation for PRIQTY for tablespaces and indexes

The calculation in Figure 57 has three variables:

- ROW_SIZE
- PERCENT_FREE
- NUM_ROWS

The ROW_SIZE and PERCENT_FREE for each tablespace and the indexes are in the tables provided for each DB2 table. To calculate the number of rows for each tablespace and index, see CICS tables and indexes: CIUCICS1 and CIUCICSX.

However, until you know how many EXEC CICS statements are used by the applications, it is difficult to estimate the number of rows for a DB2 table.

The DB2 table update jobs, CIUUPDB1, CIUUPDB2, CIUUPDB3, CIUUPDB4, CIUUPDBN, and CIUAFFLD, include a step to report on the number of rows in a table. For example:

```
--Show me the number of rows in the CIU_CICS_DATA table
***INPUT STATEMENT:
SELECT COUNT(*) FROM CIU_CICS_DATA READONLY;
+-----+
1_| 3037 |
+-----+
```

For the CIU_CICS_DATA table use the row count from the load module scanner jobs CIUJCLTS and CIUJCLTD. Or run the sample SQL member, CIUSPACE, to report on the row count for all of the CICS IA tables. Use the reported values to re-create the table space and index if required.

CICS tables and index: CIUCICS1 and CIUCICSX

How to calculate the space required for the DB2 tablespace and DB2 index for CICS data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for CICS data, estimate the total number of rows. This number is calculated from the following values; see Values required for each CICS region:

- NUM_CICS_PROG
- AVG_EXEC_SHORT
- AVG_EXEC_LONG
- PROG_TRAN_RATIO
- NUM_REGION_D

Calculate the number of rows for the CICS table as follows:

$$\begin{aligned} \text{NUM_ROWS} = & (\text{NUM_CICS_PROG} * \text{AVG_EXEC_SHORT} \\ & * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D}) + \\ & (\text{NUM_CICS_PROG} * \text{AVG_EXEC_LONG} \\ & * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D}) \end{aligned}$$

If you are using DB2 V7.1, use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 115 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 115. Worksheet for CICS tablespace using DB2 V7.1

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUCIC1	315	15		
Indexes				
XICICS11	245	20		

If you are using DB2 V8.1, use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 116 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 116. Worksheet for CICS tablespace using DB2 V8.1

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUCIC1	387	15		
Indexes				
XICICS11	317	20		

DB2 tables and index: CIUDB2

Calculation of the space required for the DB2 tablespace and DB2 index for DB2 data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for DB2 data, you need to estimate the total number of rows.

This number is calculated from the following values; see Values required for each CICS region.

- NUM_DB2_PROG
- AVG_DB2

- PROG_TRAN_RATIO
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_DB2_PROG} * \text{AVG_DB2} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 117 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the DB2 tablespace.

Table 117. Worksheet for DB2 tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUDB2D	189	15		
Indexes				
XIDB2D1	133	20		

MQ tables and index: CIUMQ1

Calculation of the space required for the DB2 tablespace and DB2 index for Websphere MQ data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for Websphere MQ data, you need to estimate the total number of rows.

This number is calculated from the following values; see Values required for each CICS region.

- NUM_MQ_PROG
- AVG_MQ
- PROG_TRAN_RATIO
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_MQ_PROG} * \text{AVG_MQ} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 118 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the MQ tablespace.

Table 118. Worksheet for MQ tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUMQD	163	15		
Indexes				
XIMQD1	107	20		

IMS tables and index: CIUIMS

Calculation of the space required for the DB2 tablespace and DB2 index for IMS data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for IMS data, you need to estimate the total number of rows.

Calculate this number from the following values; see Values required for each CICS region.

- NUM_IMS_PROG
- AVG_IMS
- PROG_TRAN_RATIO
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_IMS_PROG} * \text{AVG_IMS} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 119 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the IMS tablespace.

Table 119. Worksheet for IMS tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUMSD	127	15		
Indexes				
XIIMSD1	69	20		

Natural tables and an index: CIUNAT

Calculate the space required for the DB2 tablespace and DB2 index for Natural data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for Natural data, you estimate the total number of rows.

This number is calculated from the following values:

- NUM_NATURAL_PROG
- AVG_NATURAL
- PROG_TRAN_RATIO
- NUM_REGION_D

See Values required for each CICS region.

Calculate the number of rows for the Natural table as follows:

$$\text{NUM_ROWS} = (\text{NUM_NATURAL_PROG} * \text{AVG_NATURAL} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 120 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the DB2 tablespace.

Table 120. Worksheet for Natural tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUNATD	267	15		
Index				
XNATDUNI	124	20		

Exit resource tables and index: CIUREXIT

Calculation of the space required for the DB2 tablespace and DB2 index for the exit resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for exit resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_EXIT
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_EXIT} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 121 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 121. Worksheet for exit resource tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUREXIT	68	15		
Indexes				
XEXITA	44	20		

File resource tables and index: CIURFILE

Calculation of the space required for the DB2 tablespace and DB2 index for the file resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for file resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_FILE
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_FILE} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 122 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 122. Worksheet for file detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIURFILE	330	15		
Indexes				
XFILEA	20	20		

Program resource tables and index: CIURPROG

Calculation of the space required for the DB2 tablespace and DB2 index for the program resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for program resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_PROG
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_PROG} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 123 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 123. Worksheet for program detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUPROG	339	15		
Indexes				
XPROGA	20	20		

Transaction resource tables and index: CIURTRAN

Calculation of the space required for the DB2 tablespace and DB2 index for the transaction ID resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for transaction resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_TRAN
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_TRAN} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 124 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 124. Worksheet for the TRANSID detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUTRANS	211	15		
Indexes				
XTRANA	16	20		

Transient data queue resource tables and index: CIURTDQ

Calculation of the space required for the DB2 tablespace and DB2 index for the transient data queue resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for transient data queue resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_TDQ
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_TDQ} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 125 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 125. Worksheet for the TDQUEUE detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUTDQ	291	15		
Indexes				
XTDQUEA	16	20		

Temporary storage queue resource tables and index: CIURTSQ

Calculation of the space required for the DB2 tablespace and DB2 index for the temporary storage queue resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for temporary storage queue resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_TSQ
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_TSQ} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 126 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 126. Worksheet for the TSQUEUE detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUTSQ	114	15		
Indexes				
XTSQUEA	28	20		

Web services resource tables and index: CIURWEB

Calculation of the space required for the DB2 tablespace and DB2 index for the Web services resource data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 index required for Web services resource data, you need to estimate the total number of rows.

Calculate this number from the following values:

- NUM_CICS_WEBS
- NUM_REGION_D

Calculate the number of rows for the DB2 table as follows:

$$\text{NUM_ROWS} = (\text{NUM_CICS_WEBS} * \text{NUM_REGION_D})$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 127 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333.

Table 127. Worksheet for the WEBSERV detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUWEBS	1228	15		
Indexes				
XWEBSA	44	20		

Affinity tables and indexes

How to calculate the space required for the DB2 tablespace and DB2 indexes for Affinity data.

To calculate the space, in KB, to allocate for the DB2 tablespace and the DB2 indexes required for the Affinity collection, estimate the total number of rows. In this example, there are three tables:

CIU_AFF_INDEX. This table holds the group count for all the different affinity types. The number of rows is fixed at 20.

$$\text{NUM_ROWS} = 20$$

CIU_AFF_CMD_DATA. The number of rows for this table is an estimate of the total number of commands that cause affinities for each program in each region. Calculate this number from the following values; see Table 128 on page 341.

- NUM_AFF_PROG
- AVG_AFF
- NUM_REGION_D

The number of rows for the CICS table can be calculated as follows:

$$\text{NUM_ROWS} = (\text{NUM_AFF_PROG} * \text{AVG_AFF} * \text{NUM_REGION_D})$$

CIU_AFF_GROUP_DATA. The number of this table is an estimate of how many affinity groups are required. Affinity commands stored in the CIU_AFF_CMD_DATA table are grouped by affinity type, for example TSQueue type. Assume there are four affinity commands per group.

$$\text{NUM_ROWS} = (\text{NUM_AFF_PROG} * \text{AVG_AFF} * \text{NUM_REGION_D}) / 4$$

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 128 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the Affinity tablespace.

Table 128. Worksheet for Affinity tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUAFFD				
CIU_AFF_INDEX	6	15		
CIU_AFF_CMD_DATA	61	15		
CIU_AFF_GRP_DATA	367	15		
Indexes				
X4AFFG11	10	20		
X4AFFG12	81	20		
X4AFFC11	57	20		
X4AFFC12	53	20		
X4AFFI11	6	20		
X3GRPDAT	255	20		
X1GRPDAT	10	20		
X2AFFDAT	22	20		
X1AFFDAT	42	20		

Load Module Scanner tables

The default values set by CICS IA are larger than required. To calculate the space to allocate for the DB2 tablespace and the DB2 indexes required for the Load Module Scanner, estimate the total number of rows.

In this case there are two tables:

CIU_SCAN_SUMMARY: The number of rows for this table is the number of programs that contain Affinity or Interdependency commands from all the application load modules.

CIU_SCAN_DETAIL: The number of rows for this table is the number of programs defined above multiplied by the average number of EXEC CICS commands per program.

Obtain these values by running the Load Module Scanner in report mode only; that is, running CIUJCLLS and CIUJCLLD.

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 129 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the Load Module Scanner tablespace.

Table 129. Worksheet for Load Module Scanner tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIULMSD				
Summary	86	15		

Table 129. Worksheet for Load Module Scanner tablespace (continued)

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
Detail	127	15		
Indexes				
X4LMSDA	44	20		
X4LMSDB	44	20		

CSECT Module Scanner tables

The default values set by CICS IA are larger than required. To calculate the space to allocate for the DB2 tablespace and the DB2 indexes required for the CSECT Module Scanner, estimate the total number of rows.

There are three tables in two tablespaces:

CIU_PROGRAM_INFO: The number of rows for this table is the number of programs that are in all the application load modules.

CIU_CSECT_INFO: The number of rows for this table is the number of programs defined above, multiplied by the average number of CSECTS per program.

CIU_TRNSLATORS: This table is static. It holds the program product number for compilers and translators with the corresponding names. The number is set to 50. The default values for CICS IA PRIQTY and SECQTY for this table are larger than required. Set them to the value you obtain from the calculation below:

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 130 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the CSECT Module Scanner tablespace.

Table 130. Worksheet for CSECT Module Scanner tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUCSSD				
Program	123	15		
CSECT	240	15		
Indexes				
X4CSSDA	86	20		
X4CSSDB	94	20		
X4CSSDE	8	20		

CICS region tables

The default values set by CICS IA are larger than required. To calculate the space to allocate for the DB2 tablespace and the DB2 indexes required for CICS region information, estimate the total number of rows.

In this example, the number of rows equals the number of CICS regions for which CICS IA data is being captured:

NUM of ROWS = Number of CICS regions

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 131 to calculate the PRIQTY and SECQTY as described in Figure 57 on page 333 for the CICS region tablespace.

Table 131. Worksheet for CICS region tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUREGD	192	15		
Indexes				
XIREGDB	8	20		

CICS IA plug-in for CICS Explorer Resource table: CIURESTB/X

Calculation of the space required for the DB2 tablespace and the DB2 indexes required for the CICS region information.

The default values set by CICS IA are larger than required. To calculate the space to allocate for the DB2 tablespace and the DB2 indexes required for CICS region information, estimate the total number of rows.

In this example, the number of rows calculated for the CICS table in CICS tables and indexes section can be used.

If you are using DB2 V7.1 use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 132 to calculate the PRIQTY and SECQTY as described in Calculation for PRIQTY for tablespaces and indexes for the Resource tablespace.

Table 132. Worksheet for Resource tablespace using DB2 V7.1

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIURSRC	201	15		
Indexes				
X01RES01	201	2		
X01RES02	183	2		
X01RES03	8	2		

If you are using DB2 V8.1 or later use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 133 to calculate the PRIQTY and SECQTY as described in Calculation for PRIQTY for tablespaces and indexes for the Resource tablespace.

Table 133. Worksheet for Resource tablespace using DB2 V8.1

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIURSRC	273	15		
Indexes				
X01RES01	273	2		
X01RES02	255	2		
X01RES03	8	2		

Appendix G. CICS IA security

This section contains information on how to set up RACF security for CICS IA.

CICS IA transaction security

CICS IA has no internal RACF security classes. The two main interfaces are application programs. These two interfaces are the Operations and Administration Interface driven by transaction CINT and the Eclipse-based Query Interface.

All CICS IA transactions are defined with RESSEC(NO) and CMDSEC(NO). If you want to categorize and define the IA transactions in a similar way to CICS transactions, see Table 134. It shows the CICS IA transactions and their RACF categories as described in the *CICS RACF Security Guide*. It also indicates whether the transaction runs a program that has a DB2 DBRM associated with it.

Table 134. RACF categories for CICS IA transactions

Transid	Description	Category	DB2
CINT	Drives program CIUA000C for Operation and Administration.	3	YES
CINB	Drives program CIUCINB1 for a long running task that writes the data to VSAM (see note below).	1	
CINC	Drives program CIUACM10 for the Command Flow feature.	3	

Note:

Transaction CINB is a long-running task that cannot be started from a terminal and has no terminal associated with it. If you are collecting DB2 data, the user ID under which it runs requires authorization to the SYSIBM.SYSSTMT and SYSIBM.SYSPACSTMT tables. In most cases, the CICS default User ID is used. However, in some cases it might be the PLT user ID 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.

Authorization might be given by granting the userid access to the CICS IA batch plan.

DB2 security

CICS IA uses DB2 packages to control access to the tables in CICS and batch. All the packages are bound in to both the batch plan and the CICS plan. The plan names can be defined at customization time.

CICS IA provides DB2ENTRY and DB2TRAN resource definitions to control access to the IA tables in CICS. The following DB2ENTRY resource definition is supplied.

```
DEFINE DEFINE DB2ENTRY(CICSIAD) GROUP(_groupt_)
        DESCRIPTION(CICS IA DB2 ENTRY)
        ACCOUNTREC(NONE) AUTHTYPE(USER) DROLLBACK(YES)
        PLAN(_db2p1anc_) PRIORITY(EQUAL) PROTECTNUM(0)
        THREADLIMIT(15)
        THREADWAIT(POOL)
```

| The DB2TRAN resource definition is DEFINE DB2TRAN(CINB) ENTRY(CICSIAD)
| GROUP(_groupt_).

| See "DB2 considerations" on page 51 for further information on defining DB2
| resources.

Appendix H. How IA affects performance

Running CICS IA to obtain interdependency data and affinity data incurs a performance overhead. You must understand the cost of running the individual components of CICS IA and the performance overhead of running this workload in a single region CICS IA environment.

The environment that was used to take the measurements was configured as follows:

- An LPAR with three dedicated CPUs on an IBM 2084-303 mainframe system
- IBM ESS 800 Enterprise Storage Server®
- z/OS V1.9
- CICS TS V3.2
- CICS IA V2.2.0
- DB2 V7.1
- WebSphere MQ V6.0

Performance data collection

The basic metrics for making a comparison between CICS with and without CICS IA data collection running were as follows:

- Transaction rate (ETR)
- CICS region CPU usage
- CICS internal response times

These metrics were all collected using RMF™.

Component testing

To determine the cost of individual components in CICS IA, basic COBOL programs were written. For example, to determine the cost of collecting interdependency data for file requests, a COBOL program was written that issued the following commands:

- EXEC CICS RECEIVE
- A variable number of EXEC CICS READ commands for the same VSAM record
- EXEC CICS SEND
- EXEC CICS RETURN

The number of EXEC CICS READ commands to be issued was based on an input parameter from a simulated terminal device. Because each read is for the same record, this record exists in a VSAM buffer, and hence I/O activity is avoided.

A workload consisting of 500 simulated terminals is started using TPNS. RMF is then used to determine the CPU consumed per transaction.

Performance tests were run with both 50 and 100 EXEC CICS READ commands. The difference in the CPU time consumed per transaction in these two runs is

solely associated with the 50 additional EXEC CICS READ commands. By dividing this difference by 50, the cost of a single EXEC CICS READ command can be determined.

By running these performance tests with the following CICS IA settings, the cost of an EXEC CICS READ in different environments can be compared:

- CICS IA not running
- CICS IA running with the **Files** field on the CICS Resources Option panel set to N.
- CICS IA running with the **Files** field on the CICS Resources Option panel set to Y.
- CICS IA running with the **Files** field on the CICS Resources Option panel set to D.

By comparing these four EXEC CICS READ costs, the CICS IA overhead for various **Files=x** settings can be determined.

Number of interdependency records per workload

For CICS commands, the number of interdependency records saved for a workload was based on the number of unique combinations found for a number of items.

- CICS applid
- Transaction ID
- Program name
- Command offset within the load module
- Remote SYSID, if present
- Remote name, if present
- Resource name; for example, TS queue name
- Command ID; for example, X'0402'
- Program length

The command offset in the load module is unique for each record in the dataspace. Therefore, an application that has, for example, a loop that consists of a file control command and that is performed 20 times, has a single record in the dataspace. However, an application consisting of 20 inline file control commands has 20 records in the data space, because the command offset in the load module is different for each file control command.

For DB2 commands, the number of interdependency records saved for a workload is based on the number of unique combinations found for the following items:

- CICS applid
- Transaction ID
- Program name
- Command offset within the load module
- DB2 connect ID
- DB2 command code
- DB2 section number
- DB2 statement number
- DB2 resource name
- Program length

Number of affinity records per workload

The number of affinity records saved for a workload is based on the number of unique resource names. Resource names are associated with some commands but not others.

For example, resource names are associated with ENQUEUE, DEQUEUE, and TSQ commands, but not with GETMAIN SHARED and WAIT EVENT commands. If a CICS IA exit processes a TS queue request for a TS queue that it has not encountered before, it creates a new affinity record. CICS IA does not, however, create a new affinity record for a WAIT EVENT request.

Results for interdependency data collection

No detectable additional cost was associated with switching on CICS IA data collection, with all collection options set to N.

Table 135 shows the CPU cost in microseconds of collecting interdependency data for a selection of CICS commands, using the configuration described in Appendix H, “How IA affects performance,” on page 347. Use the Large Systems Performance Reference ratios to make the necessary adjustments for your machine environment.

Table 135. CPU cost of collecting interdependency data for a selection of CICS commands, in microseconds

API command	Category	CPU cost with category option set to Y	CPU cost with category option set to D
READ	FILES	7.0	8.0
READQ TS	TS QUEUES	6.4	7.6
READQ TD	TD QUEUES	6.1	7.1
LINK	PROGRAMS	10.2	18.1

For the PROGRAMS category, CICS IA exits are called on the LINK and also on the RETURN from the LINK. Hence the CPU cost is higher than for other categories. Although the CICS IA costs are not identical for each CICS command, with the exception of the LINK command, they are similar.

DB2 and WebSphere MQ interdependency costs

Table 136 shows the CPU cost in microseconds of collecting interdependency data for DB2 and WebSphere MQ requests.

Table 136. CPU cost of collecting interdependency data for DB2 and WebSphere MQ requests, in microseconds

API request	Category	CPU cost with category option set to Y
SELECT	DB2	12
GET and PUT	MQ	10

DB2 and MQ requests run on L8 TCBs. In multiprocessor environments, work can be run concurrently on the QR and multi-L8 TCBs. With some workloads, you

might see a detectable increase in hardware instruction and data cache misses. The rate of cache misses affects the CPU costs. Thus, the figures shown in the table above might vary, depending on your workload and the machine environment.

Interdependency exits

For interdependency data collection, if a single option is set to Y or D on the **CICS Resources Options** panel, only exits associated with that option are enabled.

However, some options enable commonly used exits if selected. For example, if the **Programs** option is set to Y for interdependency data collection, XEIIN and XEIOUT exits are enabled, and are driven for every CICS command encountered. Therefore, a performance overhead is associated with driving these exits, although no CICS IA data is collected by these exits for CICS commands that are not related to programs.

See “CICS IA exits” on page 354 for further details.

Results for affinity data collection

No detectable additional cost was associated with switching on CICS IA data collection, with all collection options set to N.

Table 137 shows the CPU cost in microseconds of collecting affinity data for some CICS commands.

Table 137. CPU cost of collecting affinity data for some CICS commands, in microseconds

API command	Category	CPU cost with category option set to Y
ENQ and DEQ	ENQ, DEQ	9.2 (see note 1)
READQ TS	TS QUEUE	4.3
INQUIRE	INQUIRE, SET	3.4
POST and WAIT	WAIT	4.8 (see note 2)

Note:

1. The 9.2 microseconds CICS IA cost of an ENQ followed by a DEQ is the total CICS IA affinity collector overhead for the two CICS commands.
2. The 4.8 microseconds CICS IA cost of a POST followed by a WAIT EVENT is the total CICS IA affinity collector overhead for the two CICS commands.

Affinity exits

There is an important difference between enabling CICS IA exits for interdependency and for affinity data collection.

- For interdependency data collection, if a single option is set to Y or D on the **CICS Resources Options** panel, exits associated with that option only are enabled.
- For affinity data collection, if a single inter-transaction or transaction-system option is set to Y on the **CICS Affinities Options** panel, all the affinity data collection exits are enabled.

So, for example, if only the COLLECT STATS option is set to Y for affinity data collection, the XEIOUT exit is enabled, and this exit is driven for every CICS command encountered. Therefore, a performance overhead is associated with driving this exit, although no CICS IA data is collected by this exit.

See “CICS IA exits” on page 354 for further details.

How collection data is saved

The data collection process stores interdependency and affinity data in a data space, thus avoiding the overhead of writing the data to a VSAM data set. However, at some point this data must be externalized to VSAM data sets. Externalization is known as “saving” the collected data.

The save process occurs in these circumstances:

- Data collection is stopped.
- Data collection is paused and **Perform periodic saves** is set to Y on the **General Options** panel.
- **Perform periodic saves** is set to Y and a periodic save threshold is reached.

The threshold is based on the number of record updates occurring in the data space.

The CINB transaction is called to perform the save process.

New records in the data space

To determine when new records are stored in the data space, see “Number of interdependency records per workload” on page 348 and “Number of affinity records per workload” on page 349

Updated records in the data space

A record is updated when **Maintain usage counts** is set to Y on the **General Options** panel, and CICS IA detects a command that is already stored in the data space being issued. Note that changing **Maintain usage counts** from N to Y has a minimal effect on CICS IA performance.

Saving records to a VSAM data set

The cost of saving a single record to a local CICS IA VSAM data set is 47 microseconds.

Example for interdependency data collection

- A workload runs in a single CICS region, and consists of 10 transactions, each of which invokes a single program.
- Each of the 10 programs reads from its unique TD queue, and issues an FC WRITE to a common file.
- Assuming that interdependency data collection is running for just TD and FC data, the total number of records in the data space is 20, consisting of 10 transaction IDs, each with 2 records (one for a TD queue read and one for an FC WRITE).
- The total CPU time to save these records to a local VSAM file is 940 microseconds (20*47).

Note that normal production workloads result in many more records than in this example.

For most resources, the resource names are explicitly defined (for example, files defined by RDO or tables defined by DB2), and are therefore relatively few in number. However, with TS queues, for example, programs can dynamically create a queue name. Potentially, there might be a large number of records in the data space because each unique queue name causes a new record to be created.

Improving the performance of DB2 catalog tables

During the installation process for CICS IA, run member hlq.SCIUSQL.OUT(CIUIBM1) to define indexes on the SYSIBM.SYSPACKSTMT and SYSIBM.SYSSTMT tables. This improves the performance of retrieving the information from the catalog tables.

Following the creation of these DB2 indexes, and before running the hlq.SCIUSMP2.OUT(CIUDBNB) member to rebind the packages and plans, you might need to run the RUNSTATS DB2 utility to ensure that these indexes are used in the new DB2 plan. The required RUNSTATS parameters are:

```
RUNSTATS TABLESPACE DSNDB06.SYSPKAGE TABLE(ALL) INDEX(ALL)
RUNSTATS TABLESPACE DSNDB06.SYSPLAN TABLE(ALL) INDEX(ALL)
```

CICS IA performance figures running the CICS DB2 application

The CICS DB2 workload was run in a single address space with, and without, CICS IA interdependency data collection running.

With data collection running, only the **DB2=Y** option was set. All other options were set to N. The workload was run at four different transaction rates, and the milliseconds of CPU consumed per transaction were measured.

The CICS IA data collection overhead was measured at 2.4 milliseconds per transaction. With an average of 200 DB2 calls per transaction, the time per call is 0.012 milliseconds (or 12 microseconds per DB2 call). This 12 microsecond value matches the value obtained from the component testing for DB2 interdependency data collection, described earlier.

Table 138. CPU consumed per transaction at different transaction rates for a CICS DB2 application

Transactions per second	CPU per transaction with IA stopped	CPU per transaction with IA running	Delta per transaction	Delta per DB2 call
34	8.6ms	11.0ms	2.4ms	0.012ms
50	8.4ms	10.8ms	2.4ms	0.012ms
66	8.5ms	10.9ms	2.4ms	0.012ms
100	8.5ms	10.9ms	2.4ms	0.012ms

CICS IA performance figures running the CICS VSAM application

The CICS VSAM workload (DSW) was run in a multiple address space environment with, and without, CICS IA interdependency data collection running.

- The workload consists of five regions (two TORs, two AORs and one FOR).
- CICS IA data collection was started in the two AOR regions.
- The CICS IA data was saved to a single set of VSAM files, connected to the AORs by MRO, using a long-running mirror task in the FOR.
- With data collection running, only the **FILES=Y** option was set. All other options were set to N.
- The workload was run at four different transaction rates, and the milliseconds of CPU consumed per transaction were measured.

The CICS IA data collection overhead was measured to be 0.07 milliseconds per transaction.

Table 139 shows figures for the combined AOR regions.

Table 139. CPU consumed per transaction at different transaction rates for a CICS VSAM application

Transactions per second	CPU per transaction with IA stopped	CPU per transaction with IA running	Delta per transaction	Delta per file control call
538	0.64ms	0.71ms	0.07ms	0.007ms
723	0.65ms	0.72ms	0.07ms	0.007ms
1081	0.65ms	0.73ms	0.08ms	0.008ms
1639	0.66ms	0.73ms	0.07ms	0.007ms

With an average of 12.76 file control calls for tasks that issue file control calls, and with 75% of tasks issuing file control calls, the number of file control calls per task is 9.57 ($12.76 \times 75 / 100$). This figure is divided into the “Delta per transaction” to obtain the “Delta per file control call” value.

This 7 microsecond value matches the value obtained from the component testing for file control interdependency data collection, described earlier.

Response times

The response time percentage increase depends on the amount of application business logic and on the number of CICS, MQ, DB2, and IMS calls per task.

A task with a large amount of business logic but few CICS, MQ, DB2, and IMS calls will have a negligible response time increase. A task with a minimal amount of business logic and with many CICS, MQ, DB2, and IMS calls will have a significant response time increase.

CICS DB2 application with interdependency data collection

Internal response times showed an increase from around 24 milliseconds to 28 milliseconds, caused by the overhead of interdependency data collection for 200 SQL requests.

CICS VSAM application with interdependency data collection

Internal response times did not noticeably increase, because of the overhead of interdependency data collection for the File Control commands. The typical internal response time was 5 milliseconds.

CICS IA exits

The data collection process occurs in global user exits (GLUEs). The exits enabled depend on which data collection options are requested. All the CICS IA exits run on the same TCB as the code that calls them, so no additional CPU overhead is caused by switching TCBs to run an exit. Entry to and exit from a GLUE consumes about 500 instructions.

Table 140 shows the GLUE exits enabled for interdependency data collection option settings. Note that GLUE CIUXDUMM is always enabled.

Table 140. GLUE exits enabled for interdependency data collection option settings

CICS IA option	Exit names	GLUE exit program
PROGRAMS=Y	XEIIN, XEIOUT	CIUXnnO1
	XPCREQC	CIUXnnP1
PROGRAMS=D	XEIIN, XEIOUT	CIUXnnO1
	XPCREQC	CIUXnnP1
	XPCFTCH	CIUXnnP2
FILES=Y	XFCREQC	CIUXnnF1
FILES=D	XFCREQC	CIUXnnF1
	XFCSREQC	CIUXnnF2
TRANSACTIONS=Y	XEIIN, XEIOUT	CIUXnnO1
	XICEREQC	CIUXnnI1
TRANSACTIONS=D	XEIIN, XEIOUT	CIUXnnO1
	XPCFTCH	CIUXnnP2
TASK CONTROL=Y	XEIIN, XEIOUT	CIUXnnO1
PRESENTATION=Y	XEIIN, XEIOUT	CIUXnnO1
TS QUEUES=Y	XTSEREQC	CIUXnnS1
TS QUEUES=D	XTSEREQC	CIUXnnS1
TD QUEUES=Y	XTDEREQC	CIUXnnD1
TD QUEUES=D	XTDEREQC	CIUXnnD1
JOURNALS=Y	XEIIN, XEIOUT	CIUXnnO1
DTP=Y	XEIIN, XEIOUT	CIUXnnO1
COUNTERS=Y	XEIIN, XEIOUT	CIUXnnO1
FEPI=Y	XEIIN, XEIOUT	CIUXnnO1
WEB SERVICES=Y	XEIIN, XEIOUT	CIUXnnO1
	XWSPRROO	CIUXnnW1
WEB SERVICES=D	XEIIN, XEIOUT	CIUXnnO1
	XWSPRROO	CIUXnnW1
EXITS=Y	XEIIN, XEIOUT	CIUXnnO1

Table 140. GLUE exits enabled for interdependency data collection option settings (continued)

CICS IA option	Exit names	GLUE exit program
	XRMIIN, XRMIOU	CIUXnnR1
OTHERS=Y	XEIIN, XEIOUT	CIUXnnO1
DB2=Y	XRMIIN, XRMIOU	CIUXnnR1
MQ=Y	XRMIIN, XRMIOU	CIUXnnR1
IMS=Y	XRMIIN, XRMIOU	CIUXnnR1

The GLUE exits enabled for affinity data collection are the same, whichever affinity data collection options are set, as shown in Table 141. Note that GLUE CIUXDUMM is always enabled.

Table 141. GLUE exits enabled for affinity data collection option settings

CICS IA option	Exit name	GLUE exit program
Any option=Y	XEIOUT	CIUZnnO1
	XICEXP	CIUZnnX1
	XMEOUT	CIUZnnM1
	XBADEACT	CIUZnnB1
	XFAINTU	CIUZnnF1
	Task-related user exit	CIUZnnI1

Controlling Collector performance

Global user exits are used by CICS IA to collect resource dependencies.

Table 142 shows the global user exits used by CICS IA to collect resource dependencies.

Table 142. Global user exits used by CICS IA to collect resource dependencies

CICS commands	
Type of command	Exit point
File Control. See Note.	XFCREQC
File Data. See Note.	XFCSREQC
Program Data. See Note.	XPCFTCH
Transaction Data. See Note.	XPCFTCH
Interval Control. See Note.	XICEREQC
EXEC CICS LINK.	XPCREQC
Transient Data. See Note.	XTDEREQC
Temporary Storage Data. See Note.	XTSEREQC
All other EXEC CICS commands listed in Appendix A, "Details of dependencies and affinities collected," on page 179 that are not in one of the above categories.	XEIIN and XEIOUT

Table 142. Global user exits used by CICS IA to collect resource dependencies (continued)

Note: For a list of the EXEC CICS commands that can be monitored by the Collector, categorized by CICS domain, that is, File Control, Interval Control, Transient Data, Temporary Storage, and others, see Appendix A, "Details of dependencies and affinities collected," on page 179.	
NON-CICS commands	
Type of command	Exit point
DB2: EXEC SQL	XRMIOUT
IMS: EXEC DLI	XRMIOUT
IMS: CBLTDLI, ASMTDLI and PLITDLI	XDLIPOST
MQ: MQM	XRMIIN and XRMIOUT
Natural: ADABAS CALLS and NATURAL PROGRAM CALLS	XRMIOUT and task-related user exit program

Note: If you specify that any subset of Natural commands are to be monitored, both the CICS IA XRMIOUT exit program and task-related user exit program are called for each monitored Natural command.

Table 143 shows the global and task-related user exits used by CICS IA to collect transaction affinities.

Table 143. Global and task-related user exits used by CICS IA to collect transaction affinities

CICS commands	
Type of affinity information collected	Exit point
Updates affinity lifetime information with Business Transaction Services (BTS) activity and process information.	XBADEACT
Intercepts the intertransaction and transaction-system commands that might cause affinities.	XEIOUT
From CICS TS for z/OS V2.2 onwards, updates affinity lifetime information with LINK3270 information.	XFAINTU
Intercepts the expiration of interval control elements (ICEs) and thus deduces which affinity table elements can be deleted because they are no longer of interest.	XICEXP
Intercepts certain CICS "DFH" messages to detect terminal logoff and user signoff and thus deduce whether affinity lifetimes are of type LOGON or SIGNON.	XMEOUT
Runs at the start and end of the user task. Its purpose is to intercept the end of pseudoconversations and thus deduce whether affinity lifetimes are of type PCONV.	Task-related user exit program

Table 144 on page 357 shows the global and task-related user exits used by CICS IA to collect Command Flow data.

Table 144. Global and task-related user exits used by CICS IA to collect Command Flow data

CICS commands		
Type of command	Exit name	Exit program
EXEC CICS LINK	XEIIIN	CIUXACO3
EXEC CICS command	XEIOUT	CIUXACO2
DB2, IMS, MQ calls	XRMIIN	CIUXACR3
	XRMIOUT	CIUXACR2
START of task	TRUE	CIUXACT1

Notes:

- If you specify that a subset of CICS commands is to be monitored, the CICS IA exit programs at the corresponding exit points, shown in Table 142 on page 355 and Table 143 on page 356, are called for that subset of commands.
- If you specify that any subset of DB2, IMS, or MQ commands is to be monitored, the CICS IA XRMIOUT exit programs are called for all the DB2, IMS, and MQ commands listed in “Non-CICS API commands detected” on page 191 (not just for the subset of monitored non-CICS commands). However, the exit programs perform much less processing on the commands that the Collector has not been instructed to monitor than on those that it has been instructed to monitor.

The invocation of a global user exit program imposes an overhead on each affected command. Applications that issue many affected commands with little other processing will notice the greatest percentage increase in pathlength. Clearly, the Collector will impact the performance of the region on which it is running.

To optimize the performance of the Collector:

- Pause the collection of data during peak workloads. You can continue the collection of data when the workloads have decreased. Select option 8, **Timer Options** in the Region Configuration menu.
- Collect data for one type of command at a time, which can be of particular value for the following CICS API commands:
 - Temporary storage
 - Programs
 - Transactions

Select option 4, **CICS Options** and option 5, **DB2/IMS/MQ Options** in the Region Configuration menu. For affinities select option 7, **Affinity Options**.

- Collect data for a restricted set of transaction identifiers by specifying the prefix of those transactions. Select the **Transaction Prefix** option in the General Options menu.
- Not collecting data for applications for which you already have collected data, or you have documentation that enables you to understand the application. Updating the transaction exclude list, or the transaction and program exclude lists. For more information see “Creating your own program exclude, transaction exclude, and resource prefix lists” on page 55. When you collect the information for a given application, add the information to the exclude lists.
- Switch off the query on the SYSIBM.SYSSTMT and SYSIBM.SYSPACKSTMT tables during the data save CINB transaction, for DB2 resource collection. Set the **Inquire on DB2 resources** option in the General Options menu to N for no.

- Control the number of times the CINB transaction is called. You can control this in the following ways:
 - Switching off the **Maintain usage counts** option in the General Options menu to reduce the number of times that records are modified in the data space and subsequently saved to the VSAM data files. Switching off usage count recording reduces the cost of processing each CICS command, and reduces the workload of the save transaction, CINB.
 - Switching off the **Perform periodic saves** option in the General Options menu. No records are saved to the VSAM data files until the Collector is stopped. Switching off periodic save reduces the overhead of the background CINB transaction, but runs the risk of losing data if CICS is shut down or CINT is purged before the Collector has successfully stopped.
 - Using the **Trigger for CINB** start option in the General Options menu. This value calls CINB when a given number of updates are made. An update is performed when a new record is added to the data space, or when an existing record is modified. Existing records are modified only if the **Maintain Usage counts** option is selected. A value of 1 for the **Trigger for CINB Start** option disables the trigger mechanism, that is, the CINB is not called. Switching off the trigger mechanism reduces the overhead of the background CINB transaction, but runs the risk of losing data if CICS is shut down, or CINT is purged before the Collector has successfully stopped.
- Control the amount of TSQueue data that you collect. If your TSQueue resource name consists of a prefix and a counter, consider defining the TSQueue prefix to the resource prefix list. See “Creating a resource prefix list” on page 57. For example, this approach will replace TSQueue names DFH00001, DFH00002 .. .DFH99999 with one name, DFH+++++. You will therefore have only one entry for each CICS command that uses this TSQueue structure.

For information about how to control these options, see “Changing the Collector options” on page 90.

Appendix I. CICS IA External Interfaces

The Appendix describes available Stored Procedures and Command Flow user exit.

CICS IA provides a number of external interfaces you can use to further adjust the processes of collecting and saving data, as well as to get access to that data from one of your own applications. These interfaces include two ready-to-use Stored Procedures and customizable Command Flow user exit.

DB2 Stored Procedures

You can use the CICS IA External Interfaces to directly access different types of collected data from your application.

To learn more about CICS IA Stored Procedures, read this section. For details about stored procedures as such, refer to *DB2 Version x.1 for z/OS Administration Guide*.

CIUSPAPP Stored Procedure

With the help of this CICS IA External Interface, you can access saved dependency data directly from your application.

What is the CIUSPAPP Stored Procedure?

CIUSPAPP is a DB2 Stored Procedure that acts as dependency data query interface. It can be called from a user application with the SQL CALL statement to get the collected dependency data from CICS IA interdependency database.

Syntax

You can invoke the CIUSPAPP procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPAPP (calltype, appcode, restype, error-message, return-code);
```

Procedure parameters

There are several input parameters that enable you to manage CIUSPAPP processing and several output parameters that inform about the process completion and errors, if any.

The table below lists all CIUSPAPP parameters. You can find more information on each of them later in this section.

Table 145. CIUSPAPP parameters

Parameter name	Input/Output	Type	Description
calltype	INPUT	CHAR(4)	Type of call
appcode	INPUT	CHAR(8)	Application code
restype	INPUT	CHAR(8)	Resource type
error-message	OUTPUT	CHAR(300)	Error message text

Table 145. CIUSPAPP parameters (continued)

Parameter name	Input/Output	Type	Description
return-code	OUTPUT	INTEGER	Return code

CIUSPAPP INPUT parameters (calltype, appcode, restype)

The **calltype** parameter specifies queries that can be called by the application. Table 2 lists all available queries and their description.

Table 146. calltype values

calltype value	Description
LIST	Calls for a list of the applications defined in CICS IA. Fetches information from the CIU_APPLS_DESC table (APPLIC_CODE and APPLIC_NAME).
CICS	Calls for a list of CICS resources. Fetches information from the CIU_CICS_DATA table (TYPE and OBJECT).
DB2	Calls for a list of DB2 resources. Fetches information from the CIU_DB2_DATA table (RESTYPE and RESNAME).
MQ	Calls for a list of MQ resources. Fetches information from the CIU_MQ_DATA table (TYPE and OBJECT).
IMS	Calls for a list of IMS resources. Fetches information from the CIU_IMS_DATA table (TYPE and OBJECT).
DEFN	Calls for a list of all transactions/programs. Fetches information from the CIU_APPLS_RESOURCE table (APPLIC_CODE, APPLIC_TYPE, and APPLIC_RESNAME).
RES	Calls for a list of CICS, DB2, IMS, and MQ resources for the defined appcode and restype parameters.

The **appcode** parameter specifies the Application Code. Required for **calltype** values CICS, DB2, MQ, IMS, DEFN, RES.

The **restype** parameter specifies the resource type. Required for the **calltype** value DEFN and has itself two values: PROGRAM and TRANSID.

CIUSPAPP OUTPUT parameters (error-message, return-code)

The **return-code** parameter contains value of the CIUSPAPP return code. Possible **return-code** values are listed in the table below.

Table 147. return-code values

Return code	Description
0	CIUSPAPP procedure completed successfully.
4	One of the following conditions exists: either the calltype value is invalid, or an SQL exception/warning occurred during the CIUSPAPP run time.
5	The appcode value is not specified.

The **error-message** parameter contains message text that describes the error or warning:

- For **return-code=4**, it provides either the relevant SQL warning, or the "Invalid call type" message, depending on the error cause.
- For **return-code=5**, it provides the following message: "Application Code must be specified".

CIUSPAPP invocation

Following is an example of a COBOL program, which calls the CIUSPAPP stored procedure and receives the contents of table CIU_APPLS_DESC:

```

IDENTIFICATION DIVISION.
PROGRAM-ID. CALLSPM.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
DATA DIVISION.
FILE SECTION.
WORKING-STORAGE SECTION.

*****
*   WORKAREAS                                     *
*****
01  WV-APPLIC-NAME          PIC X(50).
01  WV-APPLIC-CODE         PIC X(08).
01  WW-CALLTYPE            PIC X(4).
01  WW-APPLIC-CODE         PIC X(08).
01  WW-APPLIC-TYPE        PIC X(08).
01  WW-ERRMSG              PIC X(300).
01  WW-RC                  PIC S9(8) BINARY.

*****
*   A RESULT SET LOCATOR FOR THE RESULT SET THAT IS RETURNED. *
*****
01  LOC-DTLSC USAGE SQL TYPE IS
    RESULT-SET-LOCATOR VARYING.

PROCEDURE DIVISION.
*-----

MAINLINE.
    MOVE 'LIST'          TO WW-CALLTYPE.
    MOVE ' '             TO WW-APPLIC-CODE.
    MOVE ' '             TO WW-APPLIC-TYPE.

    EXEC SQL
        CALL CIUSPAPP(:WW-CALLTYPE,
                     :WW-APPLIC-CODE, :WW-APPLIC-TYPE,
                     :WW-ERRMSG, :WW-RC)
    END-EXEC.

    EXEC SQL ASSOCIATE LOCATORS (:LOC-DTLSC)
        WITH PROCEDURE CIUSPAPP
    END-EXEC.

    EXEC SQL ALLOCATE C1 CURSOR FOR RESULT SET   :LOC-DTLSC
    END-EXEC.

    EXEC SQL FETCH C1
        INTO :WV-APPLIC-CODE, :WV-APPLIC-NAME
    END-EXEC.

    DISPLAY 'CODE= ' WV-APPLIC-CODE
           'NAME= ' WV-APPLIC-NAME

    GOBACK.

```

CIUSPAFF Stored Procedure

With the help of this CICS IA External Interface, you can access saved affinity data directly from your application.

What is the CIUSPAFF Stored Procedure?

CIUSPAFF is a DB2 Stored Procedure that acts as affinity data update and query interface. It can be called from a user application with the SQL CALL statement to get the collected affinity data from CICS IA interdependency database.

Syntax

You can invoke the CIUSPAFF procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPAFF (qtype, qarg1, qarg2, rc, sqlcode, errmsg);
```

Procedure parameters

There are several input parameters that enable you to manage CIUSPAFF processing and several output parameters that inform about the process completion and errors, if any.

The table below lists all CIUSPAFF parameters. You can find more information on each of them later in this section.

Table 148. CIUSPAFF parameters

Parameter name	Input/Output	Type	Description
qtype	INPUT	CHAR(3)	Query type
qarg1	INPUT	VARCHAR(8)	First query argument
qarg2	INPUT	CHAR(10)	Second query argument
rc	OUTPUT	INTEGER	Return code
sqlcode	OUTPUT	INTEGER	SQLCODE
errmsg	OUTPUT	VARCHAR(300)	Error message text

CIUSPAFF INPUT parameters (qtype, qarg1, qarg2)

The **qtype** parameter defines which cursors the CIUSPAFF program should open on either CIU_AFF_GRP_DATA or CIU_AFF_CMD_DATA tables and return them as result sets, except for the BLD query. There are five **qtype** values, or query types, you can use: BLD, RGN, PGM, TRN, and GRP.

qarg1 and **qarg2** are query arguments you must specify for each query type.

When configuring **qtype**, consider the following:

- The *BLD* query is the only query interface that does not provide any information. Instead, it updates the tables CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA for all affinity group types and the supplied region, if it is listed in CIU_REGION_INFO table. The CICS region APPLID must be specified in **qarg1**. If wildcard mask is specified instead of the APPLID value, the CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA tables for all regions listed in the CIU_REGION_INFO table are processed.

No result set is returned.

- The *RGN* query fetches information about affinity groups of the specified type for the supplied region from the CIU_AFF_GRP_DATA table. The CICS region APPLID must be specified in the **qarg1** parameter. The affinity group type must be specified in **qarg2** as affinity GROUP ID MASK.
- The *PGM* query fetches information about affinity groups of the specified type that contain the specified program. The PROGRAM NAME must be specified in **qarg1**. The affinity group type must be specified in **qarg2** as affinity GROUP ID MASK.
- The *TRN* query fetches information about affinity groups of the specified type that contain the specified transaction. The TRANSACTION ID must be specified in **qarg1**. The affinity group type must be specified in **qarg2** as affinity GROUP ID MASK.
- The *GRP* query fetches information about all commands of the specified affinity GROUP ID in the supplied region from the CIU_AFF_CMD_DATA table.

Upon defining the **qtype** value, you should also configure the **qarg1** and **qarg2** parameters. Table 2 provides a list of matching values for the selected query type.

Table 149. Available **qarg1** and **qarg2** values

qtype value	qarg1 value	qarg2 value
BLD	APPLID	N/A
RGN	APPLID	GROUP ID MASK
PGM	PROGRAM NAME	GROUP ID MASK
TRN	TRANSACTION ID	GROUP ID MASK
GRP	APPLID	GROUP ID

Table 3 describes **qarg1** and **qarg2** value types and lengths.

Table 150. **qarg1** and **qarg2** values in detail

Query argument value	Length	Description
APPLID	8	CICS TS region APPLID.
PROGRAM NAME	≤8	CICS application program name, wildcard characters '%' allowed.
TRANSACTION ID	4	CICS application transaction ID, wildcard characters '%' allowed.
GROUP ID	10	Affinity group id.
GROUP ID MASK	10	Group mask format is 'PP:%%%%%%%%%', where PP is affinity group prefix (one of the following CW, CA, EQ, GM, GU, LD, LF, LU, RW, TS, CO, DI, EN, EX, IN, PE, RE, WA, CR, CS, UN) and '%' is a wildcard character.

CIUSPAFF OUTPUT parameters (rc, sqlcode, errmsg)

The **rc** parameter contains value of the CIUSPAFF return code. Possible **rc** values are listed in the table below.

Table 151. rc values

Return code	Description
0	CIUSPAFF procedure completed successfully.
4	CIUSPAFF procedure completed successfully, but one or more SQL warning conditions were received during SQL statements execution.
8	CIUSPAFF procedure failed due to a critical error caused by incorrect parameter values.
12	CIUSPAFF procedure failed due to a disaster error caused by SQL Exception conditions during SQL statement execution.

errmsg contains message text that describes the error or warning:

- For **rc=4**, it provides the last SQL warning message out of all SQL warnings that occurred during CIUSPAFF run time.
- For **rc=8**, it provides the invalid parameter value that caused the error. The incorrect parameter can also be found in SQLSTATE (SQLCA):
 - 99150: Invalid **qtype** value specified
 - 99155: Invalid **qarg1** value specified
 - 99160: Invalid **qarg2** value specified
- For **rc=12**, it provides SQL error message for the failed SQL statement.

The **sqlcode** parameter values depend on the return code and can be found in the table below.

Table 152. sqlcode values

Return code	sqlcode value
0	0
4	Shows SQLCODE for the last statement that caused SQL warning condition.
8	0
12	Shows SQLCODE of the failed SQL statement.

CIUSPAFF invocation

Below you can see an example of CISPAAFF invocation from COBOL program:

```

IDENTIFICATION DIVISION.
PROGRAM-ID.    CALLSP
...
DATA DIVISION.
...
01 WS-SP-QTYPE      PIC X(3).
01 WS-SP-QARG1     PIC X(8).
01 WS-SP-QARG2     PIC X(10).
01 WS-SP-ERRMSG    PIC X(300).
01 WS-SP-RETCODE   PIC S9(9) BINARY.
01 WS-SP-SQLCODE   PIC S9(9) BINARY.
...
PROCEDURE DIVISION.
...
C01-CALL-CIUSPAFF SECTION.
C01-START.
    MOVE 'RGN'      TO WS-SP-QTYPE
    MOVE 'IYDZZ420' TO WS-SP-QARG1
    MOVE 'GU.%%%%%' TO WS-SP-QARG2
    MOVE SPACES     TO WS-SP-ERRMSG

```

```

MOVE ZEROS          TO WS-SP-RETCODE
                    WS-SP-SQLCODE

EXEC SQL
    CALL CIUSPAFF(:WS-SP-QTYPE,
                 :WS-SP-QARG1,
                 :WS-SP-QARG2,
                 :WS-SP-RETCODE,
                 :WS-SP-SQLCODE,
                 :WS-SP-ERRMSG)

END-EXEC

*****
* Check if SQL CALL statement completed successfully.      *
* If not – perform corresponding action.                   *
*****
IF SQLCODE NOT = 0 AND
   SQLCODE NOT = +466 THEN
    ...
END-IF

*****
* Check if CIUSPAFF completed successfully.                *
* Act depending on CIUSPAFF RETCODE value.                 *
*****
IF WS-SP-RETCODE NOT = 0 THEN
    EVALUATE WS-SP-RETCODE
        WHEN '04'
* Process CIUSPAFF warning                                *
        ...
        WHEN '08'
* Process CIUSPAFF critical error                          *
        ...
        WHEN '12'
* Process CIUSPAFF disaster error                          *
        ...
    END-EVALUATE
*****
* Check if CIUSPAFF returns the result set and take       *
* corresponding action.                                    *
*****
IF SQLCODE = +466 THEN
    ...
END-IF
    ...
C01-EXIT.
EXIT
.

```

The CICS IA Command Flow user exit

The CICS IA Command Flow user exit provides additional options when you are saving information about the traced commands into the dataspace.

What is the CICS IA Command Flow user exit?

The CICS IA Command Flow user exit point (or the CICS IA Command Flow user exit point) is a part of the CICS IA Command Flow collector, which is used by your written program to control the information about CICS TS traced commands that is saved to the dataspace.

Besides the data written by the Command Flow collector, the CICS IA Command Flow user exit can write some additional information to the dataspace. Depending on the input data, the user exit might include or not include user-supplied data to the journal record, or might decline a journal record.

The CICS IA Command Flow collector invokes this user exit for the following CICS TS commands:

- LINK
- XCTL
- START

Note: The CICS IA Command Flow user exit runs as a part of the XEIOUT global user exit program. For additional information about CICS TS GLUE programming conventions refer to the *CICS TS Customization Guide*.

Writing the CICS IA Command Flow user exit program

The CICS IA Command Flow user exit program must be written in assembler language and must be reentrant.

Register conventions

The following register values are provided on entry to an exit program:

- Register 1 contains the address of the user exit parameter list.
- Register 13 contains the address of the standard register save area where your exit program should store its own registers immediately after being invoked.
- Register 14 contains the return address to which the exit program should branch on completion of its work. You do this by using the BR 14 instruction after restoring the calling module registers, or by using the RETURN macro.
- Register 15 contains the entry address of the exit program.

31-bit addressing implications

The implications for the Command Flow user exit program are as follows:

- The CICS IA Command Flow user exit program is invoked in 31-bit AMODE.
- The user exit can be either RMODE 24 or RMODE ANY.
- If you find it necessary to switch to 24-bit AMODE in the exit program, make sure that you return correctly in 31-bit AMODE.

The user exit parameters list

When the Command Flow user exit is invoked, the CICS IA Command Flow collector that handles the user exit provides it with a parameters list. The address of this 32-byte parameters list is passed to register 1.

The Command Flow user exit parameters list contains the following eight entries:

Parameter 1

The address of the command arguments list.

Parameter 2

The address of an 8-character application program name.

Parameter 3

The reserved parameter.

Parameter 4

The address of a 128-byte work area provided for the user exit program.

|
| **Parameter 5**

| The address of a 48-character output field 1 in which your user exit program
| must return user data 1.

| **Parameter 6**

| The address of a 48-character output field 2 in which your user exit program
| must return user data 2.

| **Parameter 7**

| The address of a 48-character output field 3 in which your user exit program
| must return user data 3.

| **Parameter 8**

| The address of a 4-byte field in which your user exit program must return
| code 0, 4 or 8.

| Possible return code values are shown in the table below.

| *Table 153. User exit return code values*

Return code	Return code meaning
0	User exit completed. Add user-supplied resource data to the journal record.
4	User exit completed. Do not include user-supplied data in the journal record.
8	User exit completed. Do not log the journal record.

| **Preparing the CICS IA Command Flow user exit program**

| The SCIUSAMP member, CIUCMDUE, contains the sample job for compiling and
| link editing the CICS IA Command Flow user exit and sample user exit programs.
| The Command Flow user exit program must be added to the RPL data set and
| defined to CICS TS with the CICS key.

| **Activating the CICS IA Command Flow user exit program**

| To activate the CICS IA Command Flow user exit program, specify the name of the
| exit load module in the **User Exit Name** field on the CICS IA Command Flow
| Options panel, CIUA01. See Figure 30 on page 106.

| **The Sample User Exit program**

| The sample user exit program gives you a basic idea about how to use the CINC
| user exit interface. It allows journal records for the XCTL, START and LINK
| commands to be written only when the EXEC CICS LINK command occurs in the
| monitored program, and only if the control is passed from a specific program.

| The logic of this program can be divided into the following steps:

- | 1. Prepare for the command for execution.
- | 2. Check whether the EXEC CICS command is a LINK command. If it is not a
| LINK command, do not write the journal record.
- | 3. Check whether the EXEC CICS LINK command was issued by the specific
| program. If it is not issued by the specific program, do not write the journal
| record.
- | 4. Fill the first User Data Area with the program name that is to be linked. Fill the
| second User Data Area with the content of COMMAREA, which was passed by
| that EXEC CICS LINK command.

| 5. Restore registers and pass control to CICS IA.
| For the sample of the user exit program see Figure 58 on page 369.
|

```

*****
| EISPLI EQU X'02'
| EISCOBOL EQU X'04'
| EISASM EQU X'08'
| COPY DFHEIPDS
| R0 EQU 0
| R1 EQU 1 PARAMETER REGISTER
| R2 EQU 2
| R3 EQU 3
| R4 EQU 4
| R5 EQU 5 CIUEPAR Base Register
| R6 EQU 6 EXEC CICS ARG List Base Register
| R7 EQU 7 EXEC CICS ARG0 Base Register
| R8 EQU 8
| R9 EQU 9
| RA EQU 10
| RB EQU 11
| RC EQU 12 BASE
| RD EQU 13 SAVE AREA
| RE EQU 14 RETURN ADDRESS
| RF EQU 15 ENTRY ADDRESS
| EJECT
| CIUUESMP CSECT
| CIUUESMP AMODE 31
| CIUUESMP RMODE ANY
| SPACE
| SAVE (14,12) SAVE REGISTERS
| BALR RC,0 LOAD BASE REGISTER
| USING *,RC
| LR R5,R1 LOAD PARAMETER LIST ADDRESS TO R5
| LR RB,RD
| USING CIUEPAR,R5 BASE USER EXIT PARMLIST
| L RD,MT_UE_WKAA LOAD WORK AREA ADDRESS TO R13
| ST RB,4(,RD) SAVE OLD SAVE AREA ADDRESS IN WORK AREA
| SPACE
| DROP R1 EIA WAS BASED ON R1 IN COPYBOOK
| L R6,MT_UE_ARGL LOAD ADDRESS OF EXEC CICS ARGS LIST
| USING EIA,R6 BASE ADDRESSES OF EXEC CICS ARGS
| L R7,EIAARG0 LOAD ARG0 ADDRESS
| USING EID,R7 BASE ARG0
| SPACE
| CLC EIDFN,LINKID LINK COMMAND?
| JNE SKIPADD NO, LEAVE
| L R2,MT_UE_PGNA LOAD ADDRESS OF PROGRAM NAME, FROM
| * WHICH EXEC CICS LINK GIVES CONTROL
| LA R3,PROGNAME LOAD ADDRESS OF REQUIRED PROGRAM NAME
| CLC 0(8,R2),0(R3) COMPARE CURRENT PROGRAM NAME WITH
| * PROGRAM NAME WE ARE INTERESTED IN.
| JNE SKIPADD LEAVE IF IT IS NOT THE DESIRED PROGRAM
| L R2,MT_UE_DAT1A
| MVC 0(19,R2),UDATA1 COPY STRING TO USER DATA FIELD ONE
| L R3,EIAARG1 ADDRESS OF ARG1 - CALLED PROGRAM NAME
| MVC 11(8,R2),0(R3) COPY IT TO USER DATA
| L R2,MT_UE_DAT2A
| MVC 0(9,R2),UDATA2 COPY STRING TO USER DATA FIELD TWO
| CLI EIDOPT2,EIDCOMM DOES COMMAREA EXIST?
| JNE NOCOMMA NO, WRITE THAT IT DOES NOT EXIST
| L R3,EIAARG2 ADDRESS OF ARG2 - PASSED COMMAREA
| MVC 11(22,R2),0(R3) PUT 22 BYTES FROM COMMAREA TO CIUUDAT2
| J PUTRC0
| NOCOMMA MVC 9(11,R2),NOCOMM COPY STRING TO USER DATA FIELD TWO
| PUTRC0 L R2,MT_UE_RETCA
| L R3,RC0
| ST R3,0(,R2) SAVE RETURN CODE RC=0
| J ENDPOINT
| SKIPADD L R2,MT_UE_RETCA
| L R3,RC8
| ST R3,0(,R2) SAVE RETURN CODE RC=8
| ENDPOINT L RD,4(,RD) RESTORE OLD SAVE AREA ADDRESS
| RETURN (14,12) RESTORE REGISTERS AND
| * RETURN TO CALLER
| EJECT
| * RETURN CODES:

```

Appendix J. Bibliography

The following books provide additional useful reference:

CICS Transaction Server

<i>CICS Application Programming Guide</i>	SC34-6433
<i>CICS Application Programming Reference</i>	SC34-6434
<i>CICS DB2 Guide</i>	SC34-6457
<i>CICS Intercommunication Guide</i>	SC34-6448
<i>CICS Resource Definition Guide</i>	SC34-6430
<i>CICS Installation Guide</i>	GC34-6426

DB2

<i>Application Programming and SQL Guide</i>	SC26-9933
<i>Administration Guide</i>	SC26-9931
<i>Utility Guide and Reference</i>	SC26-9945

Appendix K. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully

You can perform most tasks required to set up, run, and maintain your CICS system in one of these ways:

- Using a 3270 emulator logged on to CICS
- Using a 3270 emulator logged on to TSO
- Using a 3270 emulator as an MVS system console

IBM Personal Communications (Version 5.0.1 for Windows 95, Windows 98, Windows NT and Windows 2000; version 4.3 for OS/2) provides 3270 emulation with accessibility features for people with disabilities. You can use this product to provide the accessibility features you need in your CICS system.

Enabling hover help for screen readers

The CICS IA plug-in for CICS Explorer uses hover help to provide you with extra information, however tooltips might not be readily available when using a screen reader such as JAWS.

To enable JAWS to read hover help when using the CICS IA plug-in, follow the steps below:

1. Press Ctrl + Insert + number pad minus to tie the JAWS and PC cursors together. The JAWS cursor will follow the PC cursor as it moves around the screen.
2. To save this setting across sessions press Ctrl + Insert + number pad minus quickly twice.
3. If JAWS is reading out too much information when reading the hover help or application dialogues, press Ctrl + Insert + number pad minus again to turn off tooltip reading.

Index

A

- abend codes 255
- accessibility 373
- activity, affinity lifetimes 9
- add user menu 75
- affinities
 - collecting data 20
 - commands detected 192
 - functions 6
 - related CICS API commands 12
 - related CICS SPI commands 12
- Affinities Reporter 133, 138, 140
 - affinity transaction groups, modifying 139
 - CIUAFFRD job 134
 - CIUAFFRP job 134
 - output 134
 - output report (example) 135
 - overview 25
 - running 133
 - understanding the affinities 139
- affinities, understanding them 139
- affinity
 - inter-transaction 8
 - transaction-system 8
- affinity database objects
 - overview 23
- Affinity database objects
 - updating 122
- affinity lifetimes 9
 - activity 9
 - facility 9
 - logon 9
 - permanent 9
 - process 9
 - pseudoconversation 9
 - signon 9
 - system 9
- Affinity object
 - CIU_AFF_GRP_DATA table 211
- Affinity objects
 - base tables 211
 - CIU_AFF_CMD_DATA table 213
 - CIU_AFF_INDEX_DATA table 214
 - facilitating tables 214
 - structure 211
 - V_CIU_AFFINITY view 214
- affinity relations 8
 - BAPPL 8
 - global 8
 - LINK3270 8
 - LUname 8
 - user ID 8
- affinity tables and indexes
 - CICS region tables 342
 - CSECT Module Scanner tables 342
 - DB2 space allocation 340
 - Load Module Scanner tables 341
- affinity transaction group definitions, producing 137

- affinity transaction-groups, combining 164
- affinity views 214
- affinity-related commands detected by the Collector 192
- affinity-related components
 - affinities reporter 12
 - Affinities Reporter 25
 - affinity database objects 23
 - builder 12, 27
 - Load Module Scanner 6
- affinity-related components, overview 10
- affinity-related EXEC CICS commands 12
- affinity-related functions 6
- APPC mapped conversation commands without the CONVID option
 - CICS API commands 180
 - presentation commands 180
- ASMTDLI commands detected by the Collector 192
- assembler language, example, load module scanner 199
- Atomservices 118

B

- BAPPL, affinity relations 8
- base table, CICS regions 221
- base tables
 - Load Module Scanner 217
- basic affinity transaction-groups, combining 164
- bibliography 371
- BMS commands 180
- builder
 - overview 27
- Builder 159
 - changes to CIUAFFBL job 159
 - combined affinity transaction group definitions 163
 - combining basic affinity transaction groups 164
 - data sets processed report 165
 - empty transaction groups report 166
 - error report 167
 - group merge report 166
 - HEADER statements 162
 - how to run 159
 - input parameters 159
 - output 163
 - syntax for input to 160
- building the CICS IA database 50

C

- calculating CICS data space
 - space considerations 329

- calculating DB2 data space
 - space considerations 329
- calculating IMS data space
 - space considerations 329
- calculating MQ data space
 - space considerations 330
- calculating Natural data space
 - space considerations 330
- calculating resource data space
 - space considerations 330
- Calculating the space required for Interdependency Collection
 - space considerations 328
- calculation for the space required for affinity collection
 - space considerations 331
- CBLTDLI commands detected by the Collector 192
- changes for Version 2 Release 1 xx
- changes for Version 2 Release 2 xviii
- changes for Version 3 Release 1 xvii
- changes for Version 3 Release 2 xv
- changing Collector options 90
- Changing the options of the collection of data 84
- CICS API commands 180
- CICS API commands detected
 - business transaction services (BTS) commands 193
 - interval control commands 192
 - other commands 193
 - program control commands 192
 - storage control commands 192
 - task control commands 192
 - temporary storage commands 192
- CICS commands detected by the Collector 179
- CICS FEPI commands detected 190
 - CONNECTION 190
 - NODE 190
 - POOL 190
 - PROPERTYSET 190
 - TARGET 190
- cics ia command flow applid list panel, displaying 108
- cics ia command flow statistics 110
- CICS IA plug-in
 - hover help 373
- CICS IA plug-in level 174
- CICS IA security 345
- CICS region tables
 - affinity tables and indexes 342
- CICS regions base table 221
- CICS regions objects
 - CIU_REGION_INFO table 221
 - structure 221
- CICS resource definitions 61
- CICS resource objects
 - CIU_EXIT_INFO table 232
 - CIU_FILE_DETAIL table 222
 - CIU_PROGRAM_DETAIL table 224

CICS resource objects (*continued*)

- CIU_TDQUEUE_DETAIL table 228
- CIU_TRUEEXIT_INFO table 233
- CIU_TSQUEUE_DETAIL table 230
- V_CIU_TRUEEXIT_INFO table 233

CICS resource objects

- GLUE and TRUE exit resource table 232

CICS SPI commands detected 186, 193

- BRFACILITY 186
- CORBASERVER 187
- DB2ENTRY 187
- DB2TRAN 187
- DJAR 187
- FILE 188
- IPCONN 188
- JOURNALNAME 188
- JOURNALNUM 188
- JVMPROFILE 188
- LIBRARY 188
- PIPELINE 188
- PROGRAM 189
- TCPIPSERVICE 189
- TDQUEUE 189
- TRANSACTION 189
- TSMODEL 189
- TSPOOL 189
- TSQNAME 189
- TSQUEUE 189
- URIMAP 189
- WEBSERVICE 189

CINB request queue manager

- diagnostics 316

CINB transaction 20

cinc user administration 73

CINT transaction 71

CIU_SCAN_DETAIL, base table 217

CIU_SCAN_SUMMARY, base table 217

CIU_AFF_CMD_DATA, table 213

CIU_AFF_GRP_DATA, table 211

CIU_AFF_INDEX_DATA, database table 214

CIU_APPLS_DESC table, database structure 208

CIU_APPLS_RESOURCES table, database structure 209

CIU_CICS_DATA table 203

CIU_CICS_DATA, table of dependencies on CICS resources 203

CIU_CSECT_INFO, database table of dependencies on CICS resources 219

CIU_DB2_DATA table 203

CIU_DB2_DATA, table of dependencies on CICS resources 204

CIU_EVENT_DETAIL table 233

CIU_EVENT_DETAIL table 233

CIU_EXIT_INFO resource table 232

CIU_EXIT_INFO table 232

CIU_FILE_DETAIL table 222

CIU_IMS_DATA table 203

CIU_IMS_DATA, table of dependencies on CICS resources 205

CIU_MQ_DATA table 203

CIU_MQ_DATA, table of dependencies on CICS resources 206

CIU_Natural_DATA table 203

CIU_NATURAL_DATA, table of dependencies on CICS resources 207

CIU_PROGRAM_DETAIL table 224

CIU_PROGRAM_INFO, database table of dependencies on CICS resources 219

CIU_REGION_INFO, base table 221

CIU_RESOURCE table 235

CIU_SQL_DATA table 203

CIU_TDQUEUE_DETAIL table 228

CIU_THREADSAFE_CMD, file table 209

CIU_TRANSID_DETAIL table 225

CIU_TRANSLATORS, database table of dependencies on CICS resources 220

CIU_TRUEEXIT_INFO table 233

CIU_TSQUEUE_DETAIL table 230

CIU_VERSION 235

CIU_WEBSERV_DETAIL, file table 231

CIU-CHAIN table 203

CIU-CHAINP table 203

CIU000 panel, Collector Main Administration Menu example 72

CIU100 screen, Collector Operations Menu example 81

CIU150 panel, Collector Statistics Menu example 89, 173

CIU200 panel, Collector Region Configuration Menu example 91

CIU240 panel, Collector CICS Resources Options example 98

CIU245 panel, Collector CICS Resources Options example 99

CIU250 panel, Collector DB2/MQ/IMS/RMI True Resources Options example 101

CIU260 screen, Collector General Options example 94

CIU270 panel, Collector Affinities Options example 103

CIU280 screen, Collector Timer Options example 104

CIU290 panel, Collector Resource Options example 92

CIU29N screen, Collector Natural Options example 113

CIU300 panel, Collector Global Options Menu example 114

CIU400 panel, User Administration Menu example 74

 User Administration panel 74

CIU410 panel, Add User Menu

- Add User panel 75
- example 75

CIU420 panel, Copy User Menu

- Copy User Menu panel 77
- example 77

CIU440 panel, User Details Menu example 79

CIU440 panel, User Details Menu (*continued*)

- User Details Menu panel 79

CIUA01 panel, Command Flow Options

- Command Flow Options panel 106
- example 106

CIUA02 panel, Command Flow ApplID list

- Command Flow ApplID list panel 109
- example 109

CIUA03 panel, CICS IA Command Flow Statistics

- Command Flow Statistics panel 110
- example 110

CIUA04 panel, List of connected CICS regions

- example 111
- List of available CICS regions, panel 111

CIUAFF1/2/3 – jcl CIUJCLCA - dependency files

- space considerations 332

CIUAFFBL job, changes to 159

CIUAFFRD job, to run the Affinities Reporter against the Affinity database objects 134

CIUAFFRP job, to run the Affinities Reporter against the VSAM affinity data files 134

CIUCIC1/CIUCICSX - CICS tables and index

- DB2 space allocation 334

CIUCNF04 panel 46

CIUCNF05 panel 47

CIUCNF06 panel 47

CIUCNF07 panel 48

CIUDB2 - DB2 tables and index

- DB2 space allocation 334

CIUIMS - IMS tables and index

- DB2 space allocation 335

CIUINT1/2/3/4/5 – jcl CIUJCLCA - dependency files

- space considerations 332

CIUJCLCS job

- CSECT scanner 154

CIUJCLLS job, used to run the Load Module Scanner 145

CIUJCLPL, running the sample batch job 58

CIUJCLRP, Reporter job to list dependencies on resources 125

CIUJCLTD job, used to run the Load Module Scanner 151

CIUJCLTR, running the sample batch job 59

CIUJCLTS job, used to run the Load Module Scanner 147

CIUJCLXP, running the sample batch job 56

CIUJSAMP job, Tailoring the job for your environment 169

CIUJSAMP, running the job 169

CIUMQ1 - MQ tables and index

- DB2 space allocation 335

CIUNAT - Natural tables and an index

- Natural space allocation 336

- CIUREXIT - exit resource tables and index
 - DB2 space allocation 337
- CIURFILE - file resource tables and index
 - DB2 space allocation 337
- CIURPROG - program resource tables and index
 - DB2 space allocation 338
- CIURTDQ - transient data queue resource tables and index
 - DB2 space allocation 339
- CIURTRAN - transaction resource tables and index
 - DB2 space allocation 338
- CIURTSQ - temporary storage queue resource tables and index
 - DB2 space allocation 339
- CIURWEB - Web services resource tables and indexes
 - DB2 space allocation 340
- CIUSPAFF 362
- CIUSPAPP 359
- classifying a CICS IA problem 172
- cleanup job 116
- cleanup utility 115, 116
- client/server interface 24
- COBOL, example, load module scanner 199
- collecting data
 - affinity data 20
 - dependency data 19
 - how to 19, 20
- collecting data with CICS IA Explorer plug-in 118
- Collector
 - Affinities Options panel 103
 - changing options 90
 - global options 114
 - natural options 113
 - region-specific options 91, 92
 - Changing the options of the collection of data 84
 - CICS Resources Options panel 98, 99
 - CINB request queue manager diagnostics 316
 - CINB transaction 20
 - control record VSAM file 22
 - controlling 19
 - controlling performance 355
 - controlling the collection of data 80
 - DB2/MQ/IMS/RMI True Resources Options panel 101
 - dependency data VSAM files 21
 - displaying the main menu screen 72
 - displaying the state of the Collector 89
 - errors 119
 - General Options screen 94
 - general, region-specific, options 94
 - Global Options Menu panel 114
 - how affinity data is collected 20
 - how dependency data is collected 19
 - Main Administration Menu panel 72
 - Natural Options screen 113
 - Operations Menu screen 81
 - pausing data collection 85
 - Region Configuration Menu panel 91
- Collector (*continued*)
 - Resource Options panel 92
 - resuming data collection 86
 - running 71
 - saving data 20
 - specifying affinity-related CICS commands
 - affinity-related CICS commands to be monitored 102
 - specifying dependency related commands to be monitored
 - DB2, IMS, MQ commands and TRUEs to be monitored 100
 - specifying region-specific options 98
 - specifying region-specific options: general 93
 - specifying region-specific options: timers 103
 - starting data collection 81
 - Statistics Menu panel 89, 173
 - stopping data collection 87
 - table manager diagnostics 314
 - Timer Options screen 104
 - what is monitored 17
 - what is not monitored 17
- collector component overview 15
- collector main administration menu panel, displaying 72
- combined affinity transaction group definitions
 - Builder 163
- combining basic affinity transaction-groups 164
- command detected
 - atom service command 179
- command flow
 - overview 14
- command flow components 14
- command flow database objects updating 122
- command flow functions 14
- command flow statistics panel, displaying 110
- command flow user exit 365
- commands detected
 - affinity-related 12, 192
 - atomservice commands 186
 - bundle support commands 186
 - CICS API 12, 179
 - CICS API commands 192
 - CICS FEPI API 190
 - CICS FEPI SPI 190
 - CICS SPI 12, 186
 - CSD commands 186
 - DB2 191
 - dependency-related 6, 179
 - distributed transaction processing (DTP) commands 181
 - event commands 187
 - events command 181
 - exception support 181
 - file control commands 181
 - IMS 191, 192
 - interval control commands 182
 - journal control commands 182
 - JVM server commands 188
- commands detected (*continued*)
 - MQ 191
 - MQ commands 188
 - named counter server commands 182
 - Natural commands 192
 - non-CICS 191
 - other commands 185
 - program control commands 183
 - task control commands 184
 - temporary storage commands 184
 - transient data commands 184
 - WEB commands 184, 185
 - web services addressing commands 185
 - XML parser commands 185, 189
- components
 - affinity-related 10
 - command flow 14
 - dependency-related 4
- compressing affinity data 140
- configuration, data sets used 37
- configuring CICS IA Explorer plug-in 118
- contacting IBM support 255
- contacting IBM Support 176
- control record VSAM file 22, 48
- controlling Collector performance 355
- controlling the collection of dependency and affinity data 80
- copy user menu 76
- correlating Load Module Scanner and Reporter output 199
- creating a program exclude list 55
- creating a resource prefix list 57
- creating a transaction exclude list 56
- creating an include list 59
- creating log streams and GDGs 50
- creating VSAM files 30
- creating your own program exclude, transaction exclude, and resource prefix lists 55
- creating your own TRUE include list 58
- CSECT Load Module Scanner
 - how to run 153
- CSECT Load Module Scanner report example 155
- CSECT Module Scanner tables
 - affinity tables and indexes 342
- CSECT scanner
 - CIUJCLCS job 154
 - overview 27
- CSECT Scanner base tables 219
- CSECT Scanner object
 - V_CIU_CICS_LINKED view 220
 - V_CIU_CSECT_TRANS view 221
- CSECT Scanner object views 220
- CSECT Scanner objects
 - CIU_CSECT_INFO table 219
 - CIU_PROGRAM_INFO table 219
 - CIU_TRANSLATORS table 220
 - structure 219
- CSECT scanner, printed report 154
- customizing CICS IA 29, 30, 45
- customizing command flow data collection 105
- customizing DB2 environment 32

Customizing the CICS IA Natural Interface 64

D

data sets processed report, Builder 165
data sets used during configuration 37
data space allocation
 space considerations 328
database objects, Affinity
 updating 122
database objects, command flow
 updating 122
database objects, Dependency
 creating 51
 DB2 51
 structure 203
 updating 121
database structure 203
 CIU_APPLS_DESC table 208
 CIU_APPLS_RESOURCES table 209
 dependency views 210
 V_CIU_CICS_INDS 210
 V_CIU_DB2_RES 211
date formatter diagnostics 316
DB2 commands detected by the Collector 191
DB2 objects
 CIU_THREADSafe_CMD table 209
 CIU_WEBSERV_DETAIL table 231
DB2 security 345
DB2 space allocation
 affinity tables and indexes 340
 CICS tables and index -
 CIUCIC1/CIUCICSX 334
 DB2 tables and index - CIUDB2 334
 exit resource tables and index -
 CIUREXIT 337
 file resource tables and index -
 CIURFILE 337
 IMS tables and index - CIUIMS 335
 MQ tables and index - CIUMQ1 335
 program resource tables and index -
 CIURPROG 338
 space considerations 333
 temporary storage queue resource
 tables and index - CIURTSQ 339
 transaction resource tables and index -
 CIURTRAN 338
 transient data queue resource tables
 and index - CIURTDQ 339
 Web services resource tables and
 indexes - CIURWEB 340
DB2 Threadsafes table 209
 structure 209
DB2, bibliography 371
DB2, when creating the Dependency
 database objects 51
defining new applications 54
defining resources to CICS 30, 61
dependencies and affinities collected,
 details of 179
dependency base tables 203
Dependency base tables
 CIU_CICS_DATA table 203
dependency data VSAM files 21, 48
dependency database objects 23

Dependency database objects
 creating 51
 DB2 51
 structure 203
 updating 121
Dependency database objects
 structure 203
dependency facilitating tables
 structure of the database 208
Dependency objects
 CIU_DB2_DATA table 204
 CIU_IMS_DATA table 205
 CIU_MQ_DATA table 206
 CIU_NATURAL_DATA table 207
Dependency Reporter 125
 overview 24
dependency reporter output 199
dependency views
 database structure 210
dependency-related commands detected
 by the Collector 179
dependency-related components 4
dependency-related EXEC CICS
 commands 6
dependency-related functions 3
detailed report (Load Module Scanner)
 creating 148, 151
 output contents 149
 output example 149
details of dependencies and affinities
 collected 179
diagnostics
 CINB request queue manager 316
 data formatter 316
 table manager 314
displaying Collector statistics for a
 specified region 89
displaying the add user menu 75
displaying the cics ia command flow
 options 105
displaying the cics ia user
 administration 73
displaying the copy user menu 76
DLI commands detected by the
 Collector 191
dump facility 173

E

eb service resource table 231
Eclipse
 client front end to CICS IA 24
 client, front end to CICS IA 24
 graphical user interface (GUI) 24
errors
 abend codes 255
 Collector 119
 messages 255
example 54
example, CICS report 128
examples, load module scanner 199
Explorer, solving problems
 obtaining an error log 175
 obtaining configuration details 175
External Interface 53, 359, 362

F

facility, affinity lifetimes 9
file resource table 222
 CICS resource objects 222
 structure 222
Front End Programming Interface (FEPI)
 API commands detected by the
 Collector 190
 SPI commands detected by the
 Collector 190
function code values
 collector CINB 316
 collector table 314
functions
 command flow 14
 dependency-related 3

G

getting started with CICS IA 29
global, affinity relations 8
granting access to the plans and
 tables 52
grouping programs 54
grouping transactions 54
Grouping transactions and programs 54

H

HEADER statements, Builder 162
hover help
 CICS IA plug-in 373
how to use this information xiv

I

IBM Support
 information data sheet 177
 when to contact 176
 working with 176
IBM support, contacting 255
IMS commands detected by the
 Collector 191, 192
include list 58
information data sheet, for use with IBM
 Support 177
installation
 Natural support 63
installation customization program 44
 worksheet 44, 317
installation verification program
 (IVP) 67
 how to run 67
 overview 67
installing CICS resource definitions 62
inter-transaction affinity 8
Introduction 1
IVP (installation verification program)
 loading sample data 68
 overview 67
IVP application 54

J

- jcl CIUJCLCC control file
 - space considerations 331

L

- Link3270, affinity relations 8
- list of available CICS regions,
 - displaying 111
- list of connected CICS regions 111
- Load Module Scanner
 - base tables 217
 - CIUJCLLS job, modifying 145
 - CIUJCLTD job, modifying 151
 - CIUJCLTS job, modifying 147
 - creating a detailed report 148, 151
 - creating a summary report 145, 147
 - detailed report (example) 149
 - overview 26
 - specifying run time values 145, 147, 151
 - summary report (example) 147
- Load Module Scanner and Reporter,
 - correlating output 199
- Load Module Scanner objects
 - CIU_SCAN_DETAIL table 217
 - CIU_SCAN_SUMMARY table 217
 - structure 217
- load module scanner output 199
- Load Module Scanner tables
 - affinity tables and indexes 341
- load module scanner, example assembler language 199
- load module scanner, example COBOL 199
- load module scanner, examples 199
- loading sample data installation
 - verification program (IVP) 68
- loading static DB2 tables 55
- logon, affinity lifetimes 9
- LUName, affinity relations 8

M

- main administration menu panel,
 - displaying the Collector 72
- messages 255
- Messages that CICS IA can issue 255
- migrating
 - removing old definitions 60
- Migrating application definitions
 - migrating from version 3.1 61
- migrating from
 - CICS IA V3.1 60
- migrating from version 3.1
 - migrating application definitions 61
 - migrating the DB2 tables 60
 - migrating the VSAM collection datasets 61
 - migrating the VSAM control record file 61
- migrating the DB2 tables
 - migrating from version 3.1 60
- migrating the VSAM collection datasets
 - migrating from version 3.1 61

- migrating the VSAM control record file
 - migrating from version 3.1 61
- modifying affinity transaction groups 139
- modifying cleanup utility job 116
- Modifying the Exclusive Work Area (EWA) size 64
- Modifying the Natural Name List size. 64
- MQ commands detected by the Collector 191
- MVS linklist modules 66

N

- Natural 64
- Natural space allocation
 - Natural tables and an index - CIUNAT 336
- Natural support
 - preparing to use CICS IA 63
- Natural support installation 63
- non-CICS commands detected by the Collector 191

O

- objects, Affinity
 - structure 211
- objects, CICS regions
 - structure 221
- objects, CSECT Scanner
 - structure 219
- objects, Load Module Scanner
 - structure 217
- obtaining an error log
 - solving problems, Explorer 175
- obtaining configuration details
 - solving problems, Explorer 175
- output of the CSECT Load Module Scanner (example) 155
- output of the Dependency Reporter (example) 128, 131, 132
- output of the Load Module Scanner (example) 147
- overview
 - command flow feature 14
 - problem determination 171
- Overview 1

P

- pausing data collection 85
- performance, of Collector 355
- permanent, affinity lifetimes 9
- PLITDLI commands detected by the Collector 192
- plug-in
 - graphical user interface (GUI) 4, 14
- plug-in Level
 - solving problems 174
- plug-in resource table
 - structure of CICS IA database 235
- plug-in Resource table: CIURESTB/X DB2 space allocation 343

- plug-in, solving problems
 - viewing Eclipse plug-ins 175
- preparing to use CICS IA
 - Natural support 63
 - setup 37
- prerequisites 2
- Presentation commands
 - BMS commands 180
- problem determination
 - abend codes 255
 - classifying the problem 172
 - contacting IBM Support 176
 - dealing with errors 171
 - dump facility 173
 - information data sheet 177
 - messages 255
 - overview 171
 - preliminary checks 171
 - trace facility 172
- problem solving 171
- process, affinity lifetimes 9
- program dynamic analysis threadsafe report
 - threadsafe 141
- program exclude list, creating a 55
- program exclude, transaction exclude, and resource prefix lists, creating your own 55
- program table 224
 - structure 224
- pseudoconversation, affinity lifetimes 9

R

- RACF 345
- RACF considerations 65
- reason code values
 - collector CINB 316
 - collector table 315
 - date formatter 316
- removing old definitions
 - migrating 60
- removing resources 115
- Reporter
 - Affinities Reporter 25
 - CICS report (example) 128
 - CIUJCLRP job, modifying 125
 - DB2 report (example) 131
 - Dependency Reporter 24
 - how to run 125
 - IMS report (example) 132
 - MQ report (example) 132
 - overview 24
 - specifying run time values 125
 - threadsafe Reporter 25
- reports
 - affinity report, Affinities Reporter 135
 - Builder error 167
 - data sets processed 165
 - dependency, affinities, threadsafe 24
 - empty transaction groups 166
 - group merge 166
- requirements 2
- resource definition 61
- resource prefix list, creating a 57
- restarting a CICS region 32

- restarting CICS 66
- resuming data collection 86
- running cleanup utility 116
- running the Affinities Reporter 133
- running the Builder 159
- running the CIUJCDLS job 50
- running the CIUJCLCG job 50
- running the CIUJSAMP job 169
- running the Collector 71
- running the Collector for the first time 71
- running the CSECT Load Module Scanner 153
- running the Dependency Reporter 125
- running the installation and customization program 29
- running the installation verification program 33, 67
- running the IVP (installation verification program) 67
- running the Load Module Scanner Load Module Scanner
 - how to run 145
- running the report 138
- running the reporter 125, 141
- running the sample batch job CIUJCLPL 58
- running the sample batch job CIUJCLTR 59
- running the sample batch job CIUJCLXP 56
- Running the sample DB2 query 169
 - running the CIUJSAMP job 169

S

- saving data 20
- scanners
 - CSECT 27
 - Load Module 26
- Scanners
 - CSECT 153
 - Load Module 145
- setup for CICS IA
 - creating indexes for SYSIBM 31, 62
 - creating the Dependency database objects 51
 - DB2 51
 - creating the VSAM files 48
 - defining CICS resources 61
 - overview 37
 - restarting your CICS regions 66
 - tailoring your CICS startup job 65
- signon, affinity lifetimes 9
- solving problems 171, 174
- space allocation, space considerations 327
- space considerations
 - calculating CICS data space 329
 - calculating DB2 data space 329
 - calculating IMS data space 329
 - calculating MQ data space 330
 - calculating Natural data space 330
 - calculating resource data space 330
 - Calculating the space required for Interdependency Collection 328

- space considerations (*continued*)
 - calculation for the space required for affinity collection 331
 - control file: CIUCNTL 331
 - data space allocation 328
 - DB2 space allocation 333
 - dependency Files – CIUAFF1/2/3 – jcl CIUJCLCA 332
 - dependency Files – CIUINT1/2/3/4/5 – jcl CIUJCLCA 332
 - required data, CICS regions 327
 - total Interdependency data space calculation for Interdependency 331
 - Values required for each CICS region 327
 - VSAM data set allocation 331
- space considerations, space allocation 327
- SPUFI 169
- starting and stopping
 - PLT 65
- starting and stopping from the PLT 65
- starting data collection 81
- startup job, CICS 65
- stopping data collection 87
- Stored Procedure 53, 359, 362
- Stored Procedures 359
- structure of the CICS regions objects 221
- structure of the database 203
- structure of the DB2 objects 222
- summary of changes xvii
- summary report (Load Module Scanner)
 - creating 145, 147
 - output contents 146
 - output example 147
- supplied modules required in the MVS linklist 66
- system, affinity lifetimes 9

T

- table
 - CIU_EVENT_DETAIL 233
- table identifier values
 - collector table 314
- Tailoring the CIUJSAMP job for your environment 169
- taking a dump 173
- temporary storage compression 140
- temporary storage queue resource table 230
 - CICS resource objects 230
 - structure 230
- terminal control and information commands
 - CICS API commands 180
 - presentation commands 180
- the dependency reporter
 - output 128
- threadsafe Reporter
 - overview 25
- threadsafe table 55
- Threadsafe table, DB2
 - structure 209

- total Interdependency data space calculation for Interdependency space considerations 331
- trace facility 172
- tracing 172
- transaction affinities 7
 - what are they? 7
- transaction affinities lifetimes, worsening of 10
- transaction exclude list, creating a 56
- transaction group definitions, producing 137
- transaction resource table 225
 - structure 225
- transaction security 345
 - DB2 security 345
 - RACF categories for transactions 345
- transaction-system affinity 8
- transient data queue resource table 228
 - CICS resource objects 228
 - structure 228
- translator table 55
- TRUE include list, creating a 59
- type and function mapping for monitored commands
 - CICS input parameters 238

U

- updating DB2 tables 31, 62
- updating the Affinity database objects 122
- updating the command flow database objects 122
- updating the Dependency and Affinity database objects 121
- updating the Dependency database objects 121
- user administration 73
- user command flow options 108
- user details menu 78
- user ID, affinity relations 8

V

- V_CIU_AFFINITY, affinity view 214
- V_CIU_CICS_INDS, database structure 210
- V_CIU_CICS_LINKED, view 220
- V_CIU_CSECT_TRANS, view 221
- V_CIU_DB2_RES, database structure 211
- V_CIU_SCAN_TRDSAFE table 218
- V_CIU_TRUEEXIT_INFO table 233
- variables that can be passed to the installation, table 318
- version table
 - structure of CICS IA database 235
- viewing details of a command flow collector user 78
- viewing Eclipse plug-ins
 - solving problems, plug-in 175
- VSAM
 - control record file 48
 - creating the VSAM files 48
 - dependency data file 48

VSAM data set allocation
space considerations 331

W

W 231

Web service resource table

 CICS resources objects 231

 CICS resources structure 231

what this information is about xiii

who this information is for xiii

worksheet, for installation customization
program 317

worsening of transaction affinities
lifetimes 10

Notices

This information was developed for products and services offered in the U.S.A. IBM might 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 can be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right can 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 might 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.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply in 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 might not apply to you.

This publication 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 might make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such

information might be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. 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 "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

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

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

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

Other product and service names might be trademarks of IBM or other companies.

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To ask questions, make comments about the functions of IBM products or systems, or to request additional publications, contact your IBM representative or your IBM authorized reseller.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

User Technologies Department (MP095)
IBM United Kingdom Laboratories
Hursley Park
WINCHESTER,
Hampshire
SO21 2JN
United Kingdom

- By fax:
 - From outside the U.K., after your international access code use 44-1962-816151
 - From within the U.K., use 01962-816151
- Electronically, use the appropriate network ID:
 - IBMLink: HURSLEY(IDRCF)
 - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication title and order number
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.



Product Number: 5655-U86

SC34-7211-00

