

CICS Interdependency Analyzer for z/OS
Version 5 Release 3



User's Guide and Reference

CICS Interdependency Analyzer for z/OS
Version 5 Release 3



User's Guide and Reference

Note

Note: Before you use this information and the product it supports, read the information in “Notices” on page 455.

This edition applies to Version 5 Release 3 of the CICS Interdependency Analyzer for z/OS (product number 5655-Y22) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2001, 2015.

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

Contents

Figures	vii
--------------------------	------------

Tables	ix
-------------------------	-----------

Preface	xiii
--------------------------	-------------

Who this information is for	xiii
---------------------------------------	------

How to use this information	xiv
---------------------------------------	-----

What's new in CICS IA Version 5.3	xv
--	-----------

New or changed commands in CICS IA V5.3	
---	--

collector	xvi
---------------------	-----

Summary of changes for earlier versions	xix
--	------------

Changes for Version 5 Release 2	xix
---	-----

New or changed commands in CICS IA V5.2	
---	--

collector	xx
---------------------	----

Changes for Version 5 Release 1	xxiii
---	-------

New or changed commands in CICS IA V5.1	
---	--

collector	xxiv
---------------------	------

Changes for Version 3 Release 2	xxviii
---	--------

New or changed commands in CICS IA V3.2	
---	--

collector	xxix
---------------------	------

Changes for Version 3 Release 1	xxxv
---	------

Chapter 1. Overview of the CICS Interdependency Analyzer	1
---	----------

CICS IA requirements	2
--------------------------------	---

CICS IA interdependency functions	3
---	---

CICS IA dependency-related components	4
---	---

Dependency-related commands	6
---------------------------------------	---

CICS IA affinity related functions	6
--	---

What are transaction affinities?	7
--	---

CICS IA affinity-related components	10
---	----

Affinity-related commands	12
-------------------------------------	----

CICS IA Command Flow functions	14
--	----

CICS IA Command Flow components	14
---	----

The Command Flow Feature	14
------------------------------------	----

The Collector component	15
-----------------------------------	----

What the Collector can monitor	17
--	----

What the Collector does not monitor	17
---	----

Controlling the Collector	19
-------------------------------------	----

How dependency data is collected	20
--	----

How affinity data is collected	20
--	----

Saving data	20
-----------------------	----

The dependency data and affinity data VSAM files	21
--	----

The control record VSAM file	22
--	----

The Dependency database objects	23
---	----

The Affinity database objects	23
---	----

The CICS IA plug-in for CICS Explorer	24
---	----

CICS IA reports	24
---------------------------	----

The Dependency Reporter	25
-----------------------------------	----

The Affinities Reporter	25
-----------------------------------	----

The Threadsafe Reporter	25
-----------------------------------	----

The Scanner component	26
---------------------------------	----

The Load Module Scanner	26
-----------------------------------	----

The CSECT Scanner	27
-----------------------------	----

The Builder component	27
---------------------------------	----

Chapter 2. Getting Started with the CICS IA plug-in for CICS Explorer	29
--	-----------

Chapter 3. Configuring CICS IA	31
---	-----------

Prepare your CICS IA configuration environment..	31
--	----

Upgrade from an earlier release	32
---	----

Running the configuration exec.	32
---	----

Modify your settings	34
--------------------------------	----

Short and Full configurations	35
---	----

New DB2 configuration	35
---------------------------------	----

DB2 considerations	35
------------------------------	----

Configure a new DB2 environment	38
---	----

Create a new DB2 environment.	43
---------------------------------------	----

Step 1 - Creating the IA DB2 database	43
---	----

Step 2 - Binding the IA DB2 packages	44
--	----

Step 3 - Loading static DB2 tables	44
--	----

Step 4 - DB2 stored procedures set up	44
---	----

Loading the IVP data and connect to the CICS IA	
---	--

plug-in	45
-------------------	----

New collection configuration	46
--	----

CICS considerations	46
-------------------------------	----

Configuring a new collection environment	50
--	----

Creating a new collection environment	59
---	----

Run the IVP transaction to verify the installation	62
--	----

Upgrading an existing DB2 configuration	63
---	----

Configuring an existing DB2 environment	64
---	----

Creating the upgraded DB2 environment	65
---	----

Upgrading your collection configuration.	69
--	----

Configuring an existing collection environment	69
--	----

Creating the upgraded collection environment..	71
--	----

Run the IVP transaction to verify the installation	73
--	----

Post configuration tasks	73
------------------------------------	----

Creating your own program exclude, transaction	
--	--

exclude, command exclude, and resource	
--	--

compression list	73
----------------------------	----

Grouping transactions and programs into	
---	--

applications	80
------------------------	----

CICS IA Natural support	81
-----------------------------------	----

Starting and stopping CICS IA from the PLT ..	82
---	----

CICS IA supplied modules required in the MVS	
--	--

link list	83
---------------------	----

Creating your translation table	83
---	----

Chapter 4. Running the Collector	85
---	-----------

Running the Collector for the first time	85
--	----

Displaying the Collector Main Administration Menu	
---	--

panel	86
-----------------	----

Command Flow User Administration using	
--	--

transaction CINT	87
----------------------------	----

Controlling the collection of dependency and affinity data	94
Starting data collection	95
Changing the data collection options dynamically	97
Pausing the collection of data	99
Resuming the collection of data	100
Stopping the collection of data	101
Displaying Collector statistics for a specified region	102
Displaying dependency collector statistics for a specified region	104
Changing the Collector options	105
Specifying region-specific options: Region configuration	105
Specifying resource options: Region configuration	107
Specifying region-specific options: General ..	108
Specifying region-specific options: API and SPI commands to be monitored.	113
Specifying region-specific options: Application data collection	115
Specifying which dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored	116
Specifying which affinity-related CICS commands are to be monitored	118
Specifying region-specific options: timers . .	120
Specifying Natural options	122
Changing global options.	123
Managing the Command Flow collection using transaction CINC	125
Displaying the CICS IA Command Flow Options panel	125
Displaying the CICS IA Command Flow ApplID list panel	129
Displaying the CICS IA Command Flow Application data collection panel.	130
Displaying CICS IA Command Flow Statistics panel	131
Displaying the list of available CICS regions ..	132
Operating CICS IA Collection from the CICS IA Explorer plug-in	134

Chapter 5. Updating the Dependency and Affinity database objects 135

Updating the Dependency database objects . .	135
Database update procedure.	136
Updating the Affinity database objects	137
Updating the Command Flow database objects ..	137

Chapter 6. Managing your CICS IA data 139

Identifying the data that you collect	139
Identifying patterns in resource names . .	139
Identifying the data by using a collection ID	140
Managing the collected data	140
Identifying data to delete	141
Deleting by collection ID	144
Deleting TSQ or ENQ – related resources with a duplicate name prefix	145

Deleting by application ID	145
Deleting old versions of programs	146
Managing affinity data	147
Removing duplicate resources from the Collector files	147

Chapter 7. The CICS IA UDB database 151

Configure a new UDB environment	151
Configure an existing UDB environment	155
Create a CICS IA UDB database	155
Loading the IVP data into the CICS IA UDB database	156
Update the CICS IA UDB database	157
Preparing CSV files	158

Chapter 8. Running the Reporter . . . 161

Running the Dependency Reporter	161
Modifying a Dependency Reporter Job . . .	161
Output from the Dependency Reporter. . .	164
Running the Affinities Reporter	169
Requesting a report from the Affinities Reporter	170
Output from the Affinities Reporter	170
Running the affinity report	174
Compressing affinity data	176

Chapter 9. Running the Program Threadsafe report 177

Analyzing a sample threadsafe report	178
--	-----

Chapter 10. Running the Load Module Scanner. 183

Creating a summary report.	183
Creating a summary report with DB2 output	185
Creating a detailed report	186
Contents of a detailed report	187
Creating a detailed report with DB2 output ..	189

Chapter 11. Running the CSECT Scanner. 191

The CIUJCLCS job.	192
The CIUUDBCS job	192
Contents of the printed report.	193

Chapter 12. Running the Builder . . . 197

Editing the CIUAFFBL job	197
Syntax for input to the Builder	198
HEADER statements	200
Output from the Builder.	201
Combined affinity-transaction-group definitions	201
Data sets processed report	203
Empty transaction groups report	204
Group merge report	204
Error report	205

Chapter 13. Running the sample DB2 query 207

The CIUJSAMP job	207
Tailoring the job for your environment . . .	207

Running SPUFI.	207	Affinity facilitating table.	271
Chapter 14. Solving problems	209	Affinity views	272
Overview of CICS IA problem determination.	209	The structure of the Load Module Scanner objects	275
Dealing with errors	209	Load Module Scanner base tables	275
Collector errors.	209	The structure of the CSECT Scanner objects	277
Preliminary checks	210	CSECT Scanner base tables.	277
Classifying the problem	210	CSECT Scanner object views	279
Supplying a CICS IA trace	211	The structure of the CICS regions objects	279
Taking a dump of CICS IA	211	CICS regions base table	280
CICS IA plug-in for CICS Explorer level	213	The structure for the Detailed resource objects	281
Obtaining an error log	213	Connections table	281
Obtaining configuration details	213	File resource table	282
Viewing Eclipse plug-ins	214	Program resource table	284
Contacting IBM Support.	214	Transaction resource table	286
When to contact the Support Center.	214	Transient Data queue resource table	289
Working with the Support Center	215	Temporary Storage queue resource table	291
Information data sheet	215	Web service resource table	292
Appendix A. Details of dependencies and affinities collected	217	GLUE and TRUE exit resource table.	293
Commands monitored for potential dependencies	217	Event table	294
CICS commands detected	217	The structure of the CICS IA plug-in for CICS	
CICS FEPI commands detected	236	Explorer resource objects	296
Non-CICS API commands detected	237	The structure of the Version objects	297
Commands monitored for potential affinities.	239	The structure of the Command Flow table objects	297
CICS API commands	239	Type and Function mapping for monitored	
CICS SPI commands	240	commands	301
Details of what is detected	240	Appendix D. Messages and codes .. 319	
ENQ/DEQ	241	Contacting IBM Support.	319
TS commands	241	Messages that CICS IA can issue	319
LOAD HOLD/RELEASE	241	Collector table manager diagnostics	383
RETRIEVE WAIT/START	242	Function code values	383
ADDRESS CWA	242	Table identifier values	383
GETMAIN SHARED/FREEMAIN and		Reason code values	384
GETMAIN64 SHARED/FREEMAIN64	242	Collector CINB request queue manager diagnostics	385
LOAD/FREEMAIN	243	Function code values	385
CANCEL/DELAY/POST/START.	243	Reason code values	385
SPI commands	244	Date formatter diagnostics	385
WAIT commands	245	Reason code values	385
Appendix B. Correlating Load Module Scanner and Dependency Reporter output to source	247	Appendix E. CICS IA space considerations	387
Dependency Reporter output	247	Required data	387
Load Module Scanner output	247	Data space allocation	388
Load Module Scanner: Assembler language example	247	Calculating the space required for interdependency collection	388
Load Module Scanner: COBOL example	247	Calculating the space required for affinity collection	391
Appendix C. The structure of the CICS IA database	251	VSAM data set allocation	391
The structure of the Dependency database objects	251	Control file: CIUCNTL	391
Dependency base tables	251	Dependency files: CIUINT1, 2, 3, 4, 5, 6, 7.	392
Dependency facilitating tables.	258	Affinity files: CIUAFF1,2,3	392
Threadsafe table	259	Application file: CIUAPPL	393
Dependency views	261	DB2 space allocation	393
The structure of the Affinity objects	269	CICS tables and index: CIUCICS1 and CIUCICSX	394
Affinity base tables	269	DB2 tables and index: CIUDB2	395
		MQ tables and index: CIUMQ1	395
		IMS tables and index: CIUIMS	396
		Natural tables and an index: CIUNAT	396
		Exit resource tables and index: CIUREXIT	397

File resource tables and index: CIURFILE . . .	397	CIUSPAFF Stored Procedure	411
Program resource tables and index: CIURPROG	398	CIUSPDPG Stored Procedure	416
Transaction resource tables and index:		CIUSPTSR Stored Procedure	418
CIURTRAN	398	CIUSPAP1 Stored Procedure	422
Transient data queue resource tables and index:		CIUSPPUR Stored Procedure	425
CIURTDQ	399	Stored Procedures for Application Deployment	430
Temporary storage queue resource tables and		The CICS IA Command Flow user exit	438
index: CIURTSQ	399		
Web services resource tables and index:		Appendix H. Collecting dynamic	
CIURWEB	400	COBOL calls.	443
Affinity tables and indexes	400		
Load Module Scanner tables	401	Appendix I. Accessibility	445
CSECT Module Scanner tables.	402	Enabling hover help for screen readers	445
CICS region tables.	403		
CICS IA plug-in for CICS Explorer Resource		Index	447
table: CIURESTB/X	403		
		Notices	455
Appendix F. CICS IA security.	405	Trademarks	456
CICS IA transaction security	405	Terms and conditions for product documentation	456
DB2 security.	405		
Appendix G. CICS IA External			
Interfaces	407		
DB2 Stored Procedures	407		
CIUSPAPP Stored Procedure	407		

Figures

1. The Collector structure of CICS IA components	4	35. List of available CICS regions , CIUA04	133
2. The reporting structure of CICS IA dependency-related components	5	36. Records before the selected LAST_RUN date.	141
3. The reporting structure of the CICS IA affinity-related components	11	37. Query to show program versions and FIRST_RUN timestamps..	142
4. Command Flow option structure	15	38. Query to show all distinct programs by latest FIRST_RUN timestamp.	142
5. Collector components	16	39. Query to show latest program and program lengths.	143
6. Example program exclude list	74	40. Query showing commands to be deleted.	143
7. Example transaction exclude list	75	41. Query to delete commands from the CICS table for old commands.	144
8. Example command exclude list	76	42. The CICS IA plug-in Collection IDs window	145
9. Example of the compression rule format version 1	78	43. Example clean up utility report	150
10. Example of the compression rule format version 2	79	44. Example CICS report from the Dependency Reporter—header page	165
11. Example of the compression rule format version 3	80	45. Example CICS report from the Dependency Reporter—main body	166
12. Collector Main Administration Menu panel, CIU000	86	46. Example DB2 report from the Dependency Reporter	167
13. User Administration Menu panel, CIU400	88	47. Example MQ report from the Dependency Reporter	168
14. Add User Menu panel, CIU410	89	48. Example IMS report from the Dependency Reporter	169
15. Copy User Menu panel, CIU420.	91	49. A sample report output by the Affinities Reporter	171
16. User Details Menu panel, CIU440	93	50. Sample basic affinity-transaction-group definitions	174
17. Collector Operations Menu screen, CIU100	95	51. Example threadsafe report, header page	179
18. Collector Statistics Menu panel, CIU150	103	52. Example threadsafe report, main body	179
19. Collection Statistics Menu panel, CIU151	104	53. Example of a summary report produced by the Load Module Scanner	185
20. Collector Region Configuration Menu panel, CIU200	106	54. Builder input syntax	200
21. Collector Resource Options panel, CIU290	107	55. Sample definitions for combined affinity transaction groups	201
22. Collector General Options panel, CIU260	109	56. Sample data sets processed report.	204
23. Collector CICS Resources Options panel, CIU240.	114	57. Example empty Tranguroups report	204
24. Collector CICS Resources Options panel, CIU245.	115	58. Sample group merge report.	205
25. Application Data Collection Options screen, CIU210.	116	59. Sample error report	206
26. Collector DB2/MQ/IMS/CPSM Resource Options panel, CIU250	117	60. Collector Statistics Menu panel, CIU150, showing the CICS IA data space name	211
27. Collector Affinities Options panel, CIU270	119	61. Example output from an MVS DISPLAY ACTIVE command, showing a data space name of 00000INT	212
28. Collector Time and Dates Options screen, CIU280	121	62. Example of finding an EXEC CICS command from the argument zero	248
29. Collector Natural Options screen, CIU29N	122	63. The sample user exit program	441
30. Collector Global Options Menu panel, CIU300	123		
31. Command Flow Options panel, CIUA01	126		
32. Command Flow ApplID list panel, CIUA02	129		
33. Command Flow Application Data Collection panel, CIUA0A	131		
34. CICS IA Command Flow Statistics panel, CIUA03	132		

Tables

1. Affinity-related CICS API and SPI commands detected by the CICS IA Collector and the CICS IA Load Module Scanner	13	39. View V_CIU_DB2_TRANSID_DET	265
2. The control keys on the User Administration Menu panel	88	40. View V_CIU_MQ_TRANSID_DET	265
3. The control keys on the Add User Menu panel	90	41. View V_CIU_IMS_TRANSID_DET	266
4. The control keys on the Copy User Menu panel.	92	42. View V_CIU_CICS_WEBSERV	267
5. The control keys on the User Details Menu panel.	94	43. View V_CIU_DB2_WEBSERV	267
6. Methods for starting data collection by the Collector	95	44. View V_CIU_MQ_WEBSERV	268
7. Methods for changing data collection options	98	45. View V_CIU_IMS_WEBSERV	268
8. Methods for pausing data collection by the Collector	99	46. The CIU_AFF_GRP_DATA table	269
9. Methods for resuming data collection by the Collector	100	47. The CIU_AFF_CMD_DATA table	271
10. Methods for stopping data collection by the Collector	101	48. The CIU_AFF_INDEX_DATA table	272
11. The User Command Flow Utility control keys on the Command Flow Options panel	128	49. View V_CIU_AFFINITY	272
12. The User Command Flow Utility control keys on the Command Flow ApplID list panel	130	50. The CIU_SCAN_SUMMARY table	275
13. The User Command Flow Utility control keys on the User Command Flow Statistics panel	132	51. The CIU_SCAN_DETAIL table	275
14. The User Command Flow Utility control keys on the CIUA04 panel	133	52. The V_CIU_SCAN_TRDSAFE table	276
15. Relationships between the CSV files and the corresponding DB2 tables and the sample jobs that create the CSV files.	159	53. The CIU_PROGRAM_INFO table	277
16. Resulting affinity relations	202	54. The CIU_CSECT_INFO table	278
17. Resulting affinity lifetimes (LUNAME relation)	202	55. The CIU_TRANSLATORS table	279
18. Resulting affinity lifetimes (BAPPL relation)	203	56. View V_CIU_CICS_LINKED	279
19. Resulting affinity lifetimes (USERID relation)	203	57. View V_CIU_CSECT_TRANS	279
20. Resulting affinity lifetimes (LINK3270 relation)	203	58. The CIU_REGION_INFO table	280
21. Resulting affinity lifetimes (GLOBAL relation)	203	59. The CIU_CONNECTIONS table	281
22. Data sheet of problem determination information for IBM Support Center	215	60. The CIU_FILE_DETAIL table	282
23. The CIU_CICS_DATA table	251	61. The CIU_PROGRAM_DETAIL table	284
24. The CIU_DB2_DATA table	253	62. The CIU_TRANSID_DETAIL table	286
25. The CIU_IMS_DATA table	254	63. The CIU_TDQUEUE_DETAIL table	289
26. The CIU_MQ_DATA table	255	64. The CIU_TSQUEUE_DETAIL table	291
27. The CIU_NATURAL_DATA table	256	65. The CIU_WEBSERV_DETAIL table	292
28. The CIU_APPLS_DESC table	258	66. The CIU_EXIT_INFO table	293
29. The CIU_APPLS_RESOURCES table	258	67. The CIU_TRUEEXIT_INFO table	294
30. The CIU_CICS_CHAINP table	258	68. The V_CIU_TRUEEXIT_INFO table	294
31. The CIU_CICS_CONNP table	259	69. The CIU_EVENT_DETAIL table	294
32. The CIU_THREADSafe_CMD table	259	70. The CIU_RESOURCE table	296
33. View V_CIU_DB2_RES	261	71. CIU_VERSION table	297
34. View V_CIU_CICS_INDS2	261	72. The CIU_CMDFLOW_DATA table	297
35. View V_CIU_DB2_INDS2	262	73. The CIU_CMDFLOW_INDEX table	300
36. View V_CIU_MQ_INDS2	263	74. Type and Function mapping for monitored commands using the API ATOMServices CICS resource option flag	302
37. View V_CIU_IMS_INDS2	263	75. Type and Function mapping for monitored commands using the SPI ATOMServices CICS resource option flag	302
38. View V_CIU_CICS_TRANSID_DET	264	76. Type and Function mapping for monitored commands using the SPI BRFacility CICS resource option flag	302
		77. Type and Function mapping for monitored commands using the SPI Bundles CICS resource option flag	302
		78. Type and Function mapping for monitored commands using the SPI Corbaserver CICS resource option flag	303
		79. Type and Function mapping for monitored commands using the API Counters CICS resource option flag	303
		80. Type and Function mapping for monitored commands using the SPI CSD CICS resource option flag	303

81. Type and Function mapping for monitored commands using the SPI DB2 CICS resource option flag	305	101. Type and Function mapping for monitored commands using the SPI Programs CICS resource option flag	310
82. Type and Function mapping for monitored commands using the SPI DJAR CICS resource option flag	306	102. Type and Function mapping for monitored commands using the API Programs CICS resource option flag	311
83. Type and Function mapping for monitored commands using the API EVENT proc CICS resource option flag	306	103. Type and Function mapping for monitored commands using the API Task Control CICS resource option flag	311
84. Type and Function mapping for monitored commands using the SPI EVENT proc CICS resource option flag	306	104. Type and Function mapping for monitored commands using the SPI TCPIPService CICS resource option flag	311
85. Type and Function mapping for monitored commands using the API Exits CICS resource option flag	306	105. Type and Function mapping for monitored commands using the API TD Queues CICS resource option flag	311
86. Type and Function mapping for monitored commands using the SPI Exits CICS resource option flag	306	106. Type and Function mapping for monitored commands using the SPI Temp Storage CICS resource option flag	312
87. Type and Function mapping for monitored commands using the FEPI API CICS resource option flag	307	107. Type and Function mapping for monitored commands using the API Transactions CICS resource option flag	312
88. Type and Function mapping for monitored commands using the FEPI SPI CICS resource option flag	307	108. Type and Function mapping for monitored commands using the SPI Transactions CICS resource option flag	312
89. Type and Function mapping for monitored commands using the SPI File CICS resource option flag	307	109. Type and Function mapping for monitored commands using the SPI Transient Data CICS resource option flag	312
90. Type and Function mapping for monitored commands using the API Files CICS resource option flag	308	110. Type and Function mapping for monitored commands using the API TS Queues CICS resource option flag	313
91. Type and Function mapping for monitored commands using the SPI IPCONN CICS resource option flag	308	111. Type and Function mapping for monitored commands using the API Web Services CICS resource option flag	313
92. Type and Function mapping for monitored commands using the API Journals CICS resource option flag	308	112. Type and Function mapping for monitored commands using the SPI Web Services CICS resource option flag	313
93. Type and Function mapping for monitored commands using the SPI Journals CICS resource option flag	308	113. Type and Function mapping for monitored commands using the API WSAddressing CICS resource option flag	314
94. Type and Function mapping for monitored commands using the SPI JVMServer CICS resource option flag	309	114. Type and Function mapping for monitored commands using the API XMLTransform CICS resource option flag	314
95. Type and Function mapping for monitored commands using the SPI Library CICS resource option flag	309	115. Type and Function mapping for monitored commands using the SPI XMLTransform CICS resource option flag	314
96. Type and Function mapping for monitored commands using the SPI MQCONN CICS resource option flag	309	116. The possible combinations of TYPE and FUNCTION values in DB2 queries	314
97. Type and Function mapping for monitored commands using the API Others CICS resource option flag	309	117. The possible combinations of TYPE and FUNCTION values in IMS queries	317
98. Type and Function mapping for monitored commands using the API Presentation CICS resource option flag	310	118. The possible combinations of TYPE and FUNCTION values in MQ queries	317
99. Type and Function mapping for monitored commands using the API Presentation or the API DTP CICS resource option flag	310	119. The possible combinations of TYPE and FUNCTION values in Natural queries	317
100. Type and Function mapping for monitored commands using the API Presentation or the API Others CICS resource option flag	310	120. Values required for each CICS region	387
		121. Values required for VSAM and DB2 calculations	388
		122. Worksheet for CICS tablespace using DB2 V7.1.	394
		123. Worksheet for CICS tablespace using DB2 V8.1.	394
		124. Worksheet for DB2 tablespace	395

125. Worksheet for MQ tablespace	396	143. calltype values.	408
126. Worksheet for IMS tablespace	396	144. return-code values	408
127. Worksheet for Natural tablespace	397	145. CIUSPAFF parameters	411
128. Worksheet for exit resource tablespace	397	146. Available qarg1 and qarg2 values	412
129. Worksheet for file detail resource table	398	147. qarg1 and qarg2 values in detail	413
130. Worksheet for program detail resource table	398	148. rc values	413
131. Worksheet for the TRANSID detail resource table	399	149. sqlcode values	413
132. Worksheet for the TDQUEUE detail resource table	399	150. CIUSPDPG parameters	417
133. Worksheet for the TSQUEUE detail resource table	400	151. Input parameters	417
134. Worksheet for the WEBSERV detail resource table	400	152. return-code values	418
135. Worksheet for Affinity tablespace	401	153. CIUSPTSR parameters	419
136. Worksheet for Load Module Scanner tablespace.	402	154. Input parameters	419
137. Worksheet for CSECT Module Scanner table space	402	155. rc values	421
138. Worksheet for CICS region tablespace	403	156. sqlcode values	421
139. Worksheet for Resource table space using DB2 V7.1	403	157. CIUSPAP1 parameters	423
140. Worksheet for Resource table space using DB2 V8.1	404	158. calltype values.	424
141. RACF categories for CICS IA transactions	405	159. return-code values	424
142. CIUSPAPP parameters	407	160. CIUSPPUR parameters	425
		161. prc values	426
		162. CIUSPEPS parameters	431
		163. CIUSPEP2 parameters	433
		164. Secondary resource type	434
		165. CIUSPEP3 parameters	435
		166. CIUSPDEP parameters	437
		167. User exit return code values	440

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 CICS IA Version 5.3

CICS IA Version 5.3 delivers a wide range of important new capabilities.

Application onboarding

CICS IA Version 5.3 introduces a number of new DB2 stored procedures, which are driven by the CICS IA plug-in. CICS IA Version 5.3 assists you to create a CICS bundle that defines the entry points for your application. This bundle can then be included in your CICS application bundle, therefore, enabling you to quickly create a CICS Transaction Server V5 cloud application and deploy it into a CICS TS V5 platform.

After you deploy this application and collect the CICS IA data, you can then identify the dependencies for this application and create a new CICS bundle or update an existing CICS bundle to import these dependencies.

Scenario-based collections

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

New CICS IA perspective

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

Default configuration

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

Database purge

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

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

Other enhancements

The following enhancements are also included in CICS IA Version 5.3:

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

New or changed commands in CICS IA V5.3 collector

The support for a number of CICS Transaction Server for z/OS commands is added to CICS IA V5.3. These CICS commands are intercepted by the dependency collector. Review the details to assess the new function and whether the changes might affect your CICS IA system.

Commands added to CICS IA V5.3

CICS IA V5.3 includes the support for the following CICS Transaction Server for z/OS commands. The commands are listed in alphabetical order. For more information, see the CICS Transaction Server for z/OS Knowledge Center.

DELETE

- CHANNEL

ENABLE GWA24

- EXIT

ENABLE GWA31

- EXIT

INQUIRE

- TSQUEUE

INQUIRE NEXT

- TSQUEUE

QUERY

- CHANNEL

REQUEST

- PASSTICKET

SET

- TSQUEUE

SIGNON

- TOKEN

START

- TRANSID

START ATTACH

- TRANSID

START BREXIT

- TRANSID

START INTERVAL

- TRANSID

START REQID INTERVAL

- TRANSID

START TRANSID CHANNEL

- CHANNEL

Summary of changes for earlier versions

The changes made in earlier versions are listed in this section.

Changes for Version 5 Release 2

CICS IA V5.2 delivers a wide range of important new capabilities.

Dependency collector optimization

For the dependency collector, you can specify that only a subset of collectable tasks is collected. The **Trigger for Task Collection** option is added to the General options panel to provide this facility.

The dependency collector now provides information about new resources or dependencies that are found for a specified region.

- The total number of any new dependencies and total number of any new resources that are found when the dependency collector runs is written to the CINT log.
- A new Collector Statistics Menu is provided to assist you to determine how much data and what type of data you have collected.

Affinity Analysis in the CICS IA Version 5.2 plug-in

In the previous version of CICS IA, affinity analysis was available using batch only. The Affinity stored procedure has changed so that you can carry out this analysis using the CICS IA Version 5.2 plug-in. These include:

- Running an affinity report.
- Running the affinity builder.
- Deploying the builder output to CICSplex SM (CPSM) Work Load Management.

Deeper threadsafe analysis

The CICS IA Version 5.2 collector is updated to capture the following information:

- CPSM API calls
- Connection details (IPIC or MRO)

The Load Module Scanner is updated to capture information about whether a program is linked as reentrant.

The CICS IA threadsafe plug-in is updated to return the number of non-threadsafe CPSM commands for each program.

The CICS IA threadsafe batch and CICS IA plug-in report are updated to show the number of non-threadsafe CPSM commands.

CICS TS platform support

In CICS TS V5.1 and later, you can package applications for deployment into a platform.

The new CICS IA Version 5.2 exits capture platform information and associate it with Dependency and Command Flow data that is collected.

The CICS IA Version 5.2 plug-in has new and updated views to support this feature.

Command Flow Visualization views

The CICS IA Version 5.2 plug-in provides the following new “time line” based views that show you the following:

- TCB mode switches.
- Switches across regions.
- Application and platform switches.

Program details view

The CICS IA plug-in provides a new view so that you can compare program attributes across regions. The view consolidates program information that is obtained from the detailed collection table and the scanner tables.

Collector exit support for CICS TS V5.3

CICS IA Version 5.3 provides CICS exits so that you can collect Dependency, Affinity, and Command Flow data in a CICS TS V5.3 region.

New or changed commands in CICS IA V5.2 collector

The support for a number of CICS Transaction Server for z/OS commands is added to CICS IA V5.2. These CICS commands are intercepted by the dependency collector. Review the details to assess the new function and whether the changes might affect your CICS IA system.

Commands added to CICS IA V5.2

CICS IA V5.2 includes the support for the following CICS Transaction Server for z/OS commands. The commands are listed in alphabetical order. For more information, see the CICS Transaction Server for z/OS Knowledge Center.

CPSM ADDRESS

- None

CPSM CANCEL

- None

CPSM CONNECT

- None

CPSM COPY

- None

CPSM CREATE

- OBJECT

CPSM DELETE

- None

CPSM DISCARD

- None

CPSM DISCONNECT

- None

CPSM EXPAND

- None

CPSM FEEDBACK

- None

CPSM FETCH

- None

CPSM GET

- OBJECT

CPSM GETDEF

- OBJECT

CPSM GROUP

- None

CPSM LISTEN

- None

CPSM LOCATE

- None

CPSM MARK

- None

CPSM ORDER

- None

CPSM PERFORM

- OBJECT

CPSM PERFORM SET

- ACTION

CPSM QUALIFY

- None

CPSM QUERY

- None

CPSM RECEIVE

- OBJECT

CPSM REFRESH

- None

CPSM REMOVE

- OBJECT

CPSM SPECIFY FILTER

- OBJECT

CPSM SET MODIFY

- None

CPSM SPECIFY VIEW

- OBJECT

CPSM TERMINATE

- None

CPSM TRANSLATE

- OBJECT

CPSM UNMARK

- None

CPSM UPDATE

- OBJECT

INVOKE

- APPLICATION

READQ

- TSQUEUE

READQ NEXT

- TSQUEUE

READQ AUX

- TSQUEUE

READQ AUX NEXT

- TSQUEUE

READQ SHARED

- TSQUEUE

READQ SHARED NEXT

- TSQUEUE

VERIFY TOKEN

- USERID

Changes for Version 5 Release 1

CICS IA V5.1 delivers a wide range of important new capabilities. Some capabilities require an APAR to be applied.

Changes for Version 5 Release 1 introduced by APAR PM82414

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

Configuration enhancements

- Assign DSNAMES for your SORT utility
- Assign your system DUMP High Level Qualifier

Command flow enhancements

- Timestamp columns added to collected data
- Individual Exclude lists

New commands collected

- COBOL DISPLAY command

Changes for Version 5 Release 1

CICS TS application support

In CICS TS V5.1, you can package applications for deployment into a platform. You can now logically define the various resources that make up a business application in CICS as a single entity and deploy these resources to CICS as a single resource. An application that is defined in this way can be managed as a single entity throughout its lifecycle, making CICS application management faster, easier, and less error prone. For more information, see Applications in the CICS TS 5.2 Knowledge Center.

The updated CICS IA V5.1 exits capture Application information and associate them with Dependency Data and Command Flow Data collected.

Improved installation and configuration

In CICS IA V5.1, the configuration execution is split into two tasks:

- Configuring the target DB2 Environment.
- Configuring the CICS regions in which you want to collect data.

Cheat sheets are added to the CICS IA plug-in for CICS Explorer® so you can select the configured output sample JCL jobs for both the DB2 and CICS configurations. The cheat sheets also guide you through the jobs that must run to complete the tasks when you use the “z/OS perspective” in the CICS Explorer.

Threadsafe analysis

The ability to run a batch job to report on the “threadsafe” readiness of a particular program or all programs in a particular region is now a DB2 Stored Procedure and can be started with the CICS IA plug-in for CICS Explorer.

DB2 data lifecycle management

You can now delete your CICS IA data by “collection id” and by CICS TS

Application definitions. A new section is provided in this information about data lifecycle management; see “Managing the collected data” on page 140.

Native SQL language stored procedures

In CICS IA V5.1, you can choose to use native SQL language stored procedures or external SQL language stored procedures. Native SQL Stored Procedures are available in DB2 V9.1 and later.

A native SQL language stored procedure is used instead of an external SQL language stored procedure, in that the source code is included within the CREATE PROCEDURE statement. The difference is in the executable procedures, and also in a richer SQL language.

When using a native SQL procedure, the entire executable is contained within DB2. The advantage of this approach is that DB2 can manage these stored procedures directly. The stored procedures run in the DBM1 address space, so there is no need to create a WLM environment to manage the procedures. Because native SQL procedures run under an enclave SRB instead of a TCB, if they are remote, they are also eligible to be run in a System z9[®] Integrated Information processor (zIIP) if one is available.

Updated exits

CICS IA V5.1 provides CICS exits so that you can collect Dependency, Affinity, and Command Flow data in a CICS TS V5.1 region.

New or changed commands in CICS IA V5.1 collector

The support for a number of CICS Transaction Server for z/OS commands is added to CICS IA V5.1. These CICS commands are intercepted by the dependency collector. Review the details to assess the new function and whether the changes might affect your CICS IA system.

Commands added to CICS IA V5.1

CICS IA V5.1 includes the support for the following CICS Transaction Server for z/OS commands. The commands are listed in alphabetical order. For more information, see the CICS Transaction Server for z/OS Knowledge Center.

ABEND

- PROGRAM

ADD

- SOAPFAULT

BIF DEEDIT

- FIELD

BUILD

- ATTACH

CANCEL

- None

CLOSE

- WEB

CHANGE

- TASK

CONVERSE

- WEB

CREATE

- DOCUMENT
- SOAPFAULT

DELAY

- None

DELETE

- DOCUMENT
- SOAPFAULT

DUMP

- TRANSID

ENTER TRACENUM

- None

EXTRACT

- ATTACH
- ATTRIBUTES
- CERTIFICATE
- LOGONMSG
- TCPIP
- TCT

FREEMAIN64

- STORAGE

GET64

- CONTAINER

GETMAIN64

- STORAGE

GETMAIN64 SHARED

- STORAGE

INQUIRE

- EPADAPTERSET

INQUIRE EPADAPTINSET

- EPADAPTER

INQUIRE NEXT

- EPADAPTERSET

INQUIRE NEXT EPADAPTINSET

- EPADAPTER

INSERT

- DOCUMENT

ISSUE

- ENDFILE
- ENDOUTPUT
- ERASEAUP
- PREPARE
- PRINT
- SEND EODS

ISSUE ABORT

- None

ISSUE ADD

- None

ISSUE END

- None

ISSUE ERASE

- None

ISSUE LOAD

- PROGRAM

ISSUE NOTE

- None

ISSUE QUERY

- None

ISSUE RECEIVE

- None

ISSUE REPLACE

- None

ISSUE SEND

- None

ISSUE WAIT

- None

MONITOR

- None

OPEN

- WEB

PARSE URL

- WEB

PERFORM

- SSL

POINT

- None

POST

- None

PUSH

- HANDLE

PUT64

- CONTAINER

RECEIVE

- PARTITIONSET

RETRIEVE

- DOCUMENT

SEND

- CONTROL
- PARTITIONSET

SET

- DOCUMENT
- EPADAPTERSET

SUSPEND

- None

WAIT

- CONVID
- EVENT
- WAIT SIGNAL

WRITE OPERATOR

- None

Changes for Version 3 Release 2

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 later.

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 data set owned by the *userid*. Now, you can save them in a shared data set.

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.

New or changed commands in CICS IA V3.2 collector

The support for a number of CICS Transaction Server for z/OS commands is added to CICS IA V3.2. These CICS commands are intercepted by the dependency collector. Review the details to assess the new function and whether the changes might affect your CICS IA system.

Commands added to CICS IA V3.2

CICS IA V3.2 includes the support for the following CICS Transaction Server for z/OS commands. The commands are listed in alphabetical order. For more information, see the CICS Transaction Server for z/OS Knowledge Center.

ACQUIRE

- TERMINAL

ADDRESS SET

- None

AP NOOP

- FEPI

ASKTIME

- None

ASKTIME ABSTIME

- TIME

CHANGE PASSWORD

- USERID

CHANGE PHRASE

- USERID

COLLECT

- STATISTICS

CONVERTTIME

- TIME

CREATE

- CONNECTION
- DB2CONN
- ENQMODEL
- JOURNALMODEL
- LSRPOOL
- MAPSET
- PARTITIONSET
- PARTNER
- PROFILE
- PROCESSTYPE

- REQUESTMODEL
- SESSION
- TERMINAL
- TRANCLASS
- TYPETERM

DISCARD

- AUTINSTMODEL
- DB2CONN
- CONNECTION
- ENQMODEL
- EPADAPTER
- JOURNALMODEL
- PARTNER
- PROCESSTYPE
- PROFILE
- REQUESTMODEL
- TERMINAL
- TRANCLASS

DISCARD NODELIST

- FEPINODE

DISCARD TARGETLIST

- FEPITARGET

ENDBROWSE

- ACTIVITY
- CONTAINER
- EVENT
- PROCESS
- TIMER

EXTRACT

- STATISTICS

FORMATTIME

- TIME

GETNEXT

- ACTIVITY
- CONTAINER
- EVENT
- PROCESS
- TIMER

HANDLE

- AID
- CONDITION

IGNORE

- CONDITION

INQUIRE

- ACTIVITYID
- ASSOCIATION
- AUTINSTMODEL
- AUTOINSTALL
- BEAN
- CAPDATAPRED
- CAPINFOSRCE
- CAPOPTPRED
- CFDTPOOL
- CLASSCACHE
- CONNECTION
- CONTAINER
- DB2CONN
- DELETSHIPED
- DISPATCHER
- DSNAME
- DUMPDS
- ENQ
- ENQMODEL
- EPADAPTER
- EVENT
- EXCI
- EXITPROGRAM
- HOST
- IPFACILITY
- IRC
- JOURNALMODEL
- JVM
- JVMPPOOL
- JVMPROFILE
- MODENAME
- MONITOR
- MVSTCB
- NETNAME
- OSGIBUNDLE
- OSGISERVICE
- PARTNER
- PROCESS

- PROCESSTYPE
- PROFILE
- REQID
- REQUESTMODEL
- RRMS OPENSTATUS
- STORAGE
- STATISTICS
- STREAMNAME
- SUBPOOL
- SYSDUMPCODE
- SYSTEM
- TASK
- TCLASS
- TCPIP
- TEMPSTORAGE
- TERMINAL
- TIMER
- TRACEDEST
- TRACEFLAG
- TRACETYPE
- TRANCLASS
- TRANDUMPCODE
- TSMODEL
- TSPOOL
- UOW
- UOWDSNFAIL
- UOWLINK
- VTAM
- WEB
- WORKREQUEST

INQUIRE NEXT

- AUTINSTMODEL
- BEAN
- CAPDATAPRED
- CAPINFOSRCE
- CAPOPTPRED
- CFDTPOOL
- CONNECTION
- DSNNAME
- ENQ
- ENQMODEL
- EPADAPTER
- EXCI
- EXITPROGRAM

- HOST
- JOURNALMODEL
- JVM
- JVMPROFILE
- MODENAME
- MVSTCB
- NETNAME
- OSGIBUNDLE
- OSGISERVICE
- PARTNER
- PROCESSTYPE
- PROFILE
- REQID
- REQUESTMODEL
- STREAMNAME
- SUBPOOL
- SYSDUMPCODE
- TERMINAL
- TRANCLASS
- TRANDUMPCODE
- TSMODEL
- TSPOOL
- UOW
- UOWDSNFAIL
- UOWLINK
- WORKREQUEST

INSTALL NODELIST

- FEPINODE

INSTALL TARGETLIST

- FEPITARGET

PERFORM

- CLASSCACHE
- DELETSHIPED
- DUMP
- ENDAFFINITY
- JVMPOOL
- SECURITY
- SHUTDOWN
- STATISTICS

PERFORM RESETTIME

- None

QUERY

- SECURITY

RELEASE

- PROGRAM

RESYNC

- ENTRYNAME

RETRIEVE

- None

SET

- AUTOINSTALL
- CLASSCACHE
- CONNECTION
- DB2CONN
- DELETSHIPED
- DISPATCHER
- DOCTEMPLATE
- DSNNAME
- DUMPDS
- ENQMODEL
- EPADAPTER
- HOST
- IRC
- JVMPOOL
- MODENAME
- MONITOR
- NETNAME
- PROCESSTYPE
- STATISTICS
- SYSDUMPCODE
- SYSTEM
- TASK
- TCLASS
- TCPIP
- TEMPSTORAGE
- TERMINAL
- TRANCLASS
- TRACEDEST
- TRACEFLAG
- TRACETYPE
- TRANDUMPCODE
- UOW
- UOWLINK

- VTAM
- WEB
- WORKREQUEST

SIGNOFF

- None

SIGNON

- USERID

SP NOOP

- FEPI

STARTBROWSE

- ACTIVITY
- CONTAINER
- EVENT
- PROCESS
- TIMER

VERIFY PASSWORD

- USERID

VERIFY PHRASE

- USERID

WAIT

- JOURNAL

WRITE

- JOURNAL

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.

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 affects the performance of your system

A new section describing how using CICS IA affects performance has been added. This section describes the performance overhead associated with collecting interdependency and affinity data.

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.

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 Version 5.3 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 V8.0 for z/OS or later. 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 Version 5.3 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 CPSM calls to CICSplex® SM resources
- EXEC SQL calls to DB2 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 217.

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”
- “Dependency-related commands” on page 6

CICS IA dependency-related components

CICS IA comprises a number of components, which divide into collecting and reporting parts. The figures show how the dependency-related components of CICS IA relate to each other.

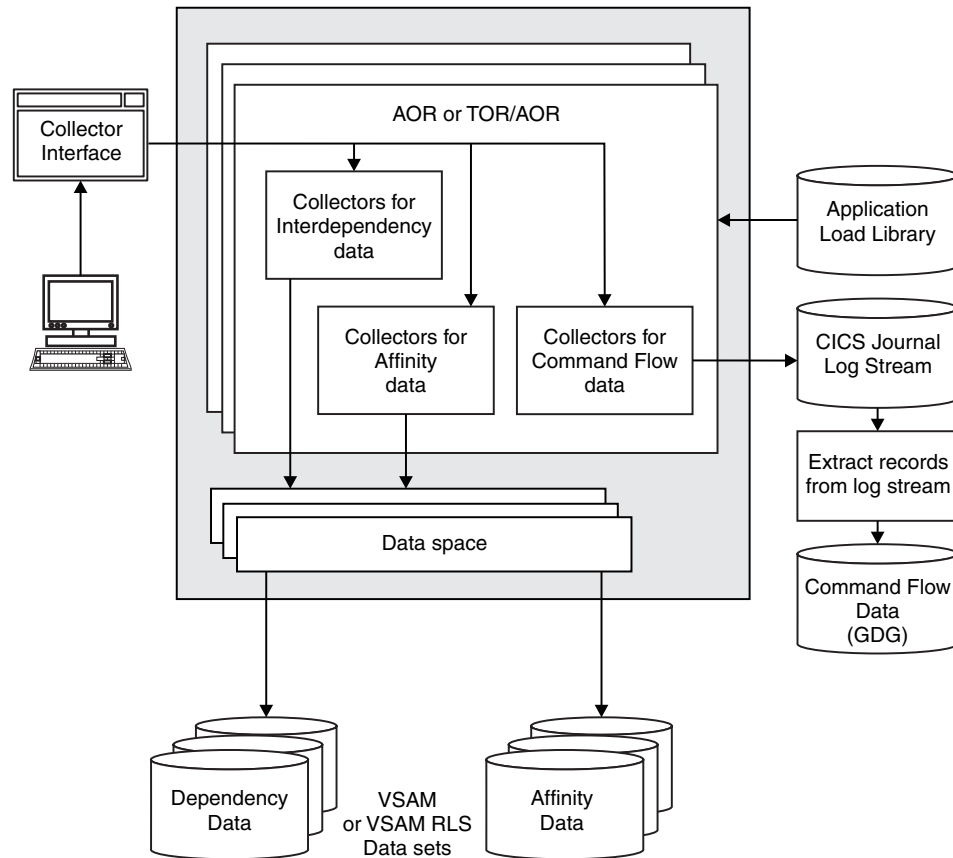


Figure 1. The Collector structure of CICS IA components

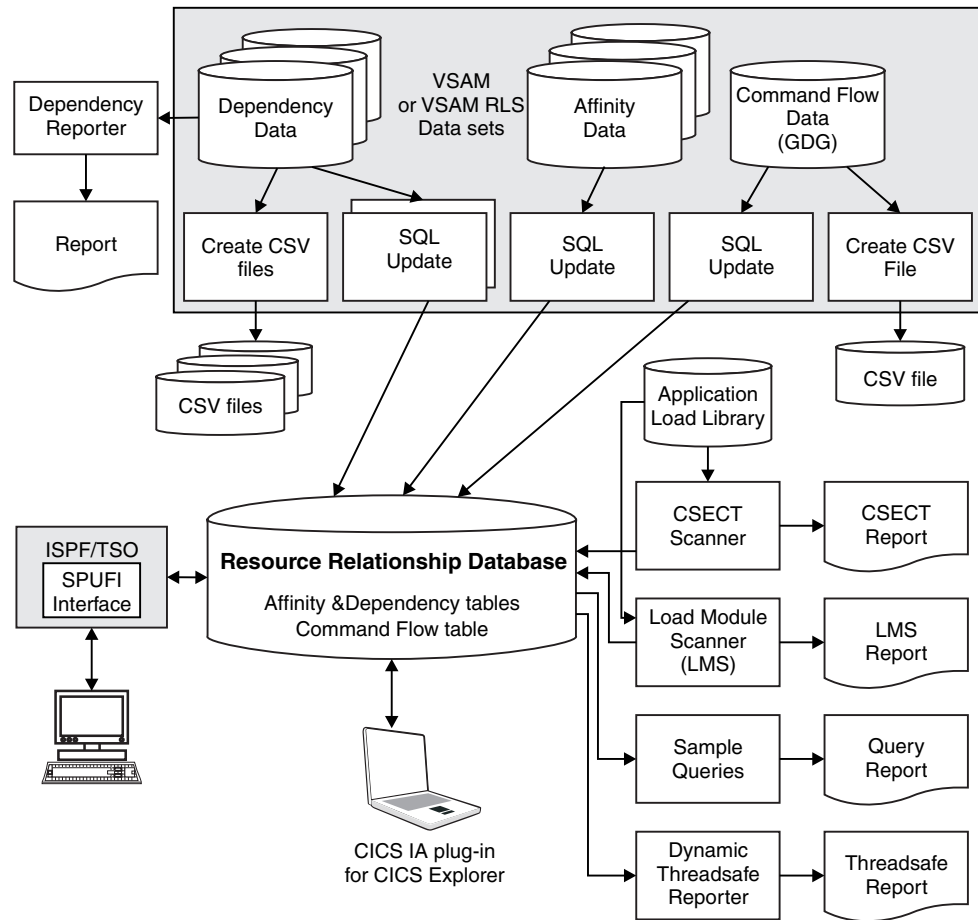


Figure 2. The reporting structure of CICS IA dependency-related components

CICS IA contains the following 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 *IBM CICS IA plug-in for 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 217.

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” 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

The figure shows how the affinity-related components of CICS IA relate to each other.

Refer to “CICS IA dependency-related components” on page 4 for the Collector structure.

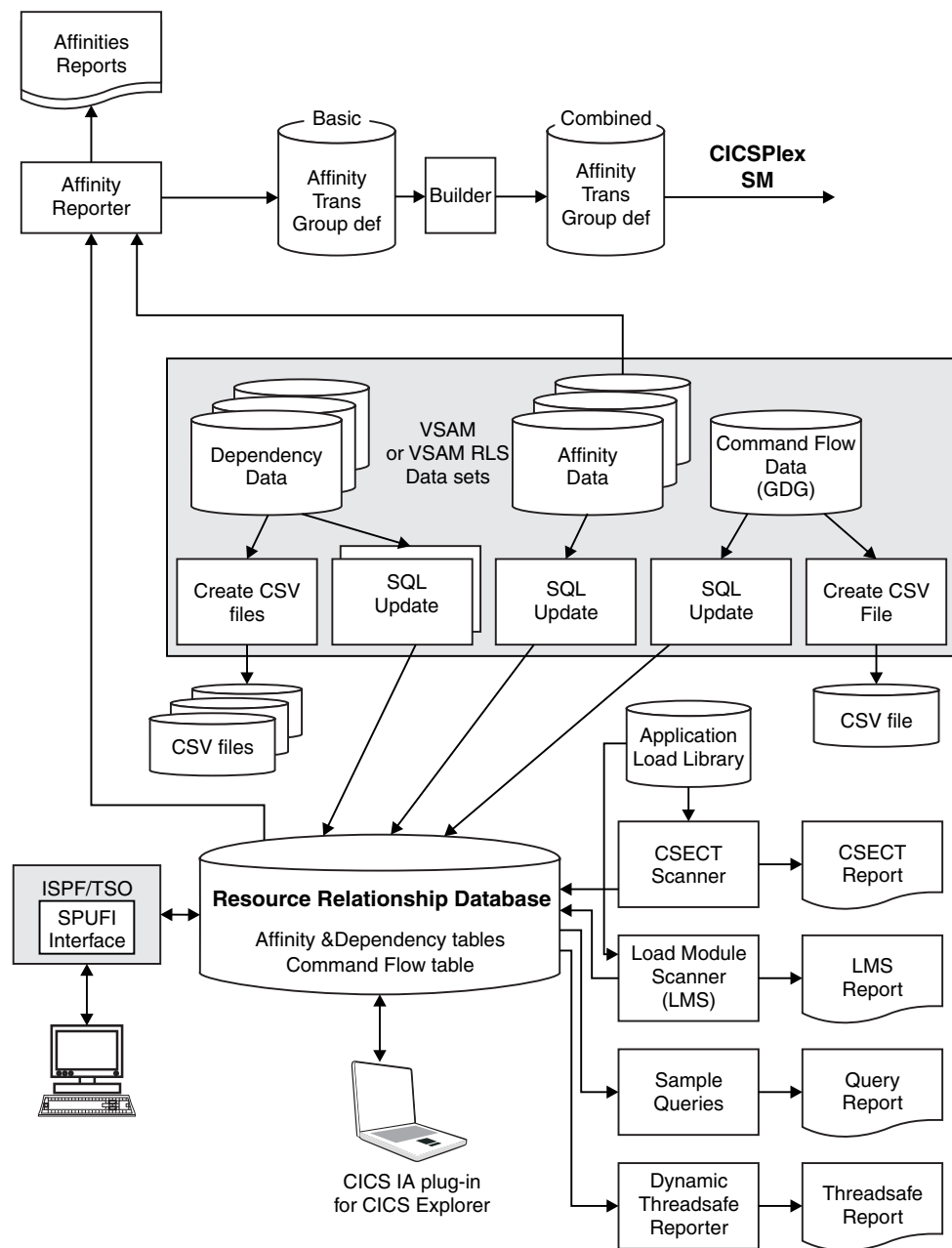


Figure 3. The reporting structure of the CICS IA affinity-related components

CICS IA includes the following 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 *IBM CICS IA plug-in for 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 239.

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

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

Refer to “CICS IA dependency-related components” on page 4 for the Collector structure.

CICS IA contains the following Command Flow components:

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 297.

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 *IBM CICS IA plug-in for 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.

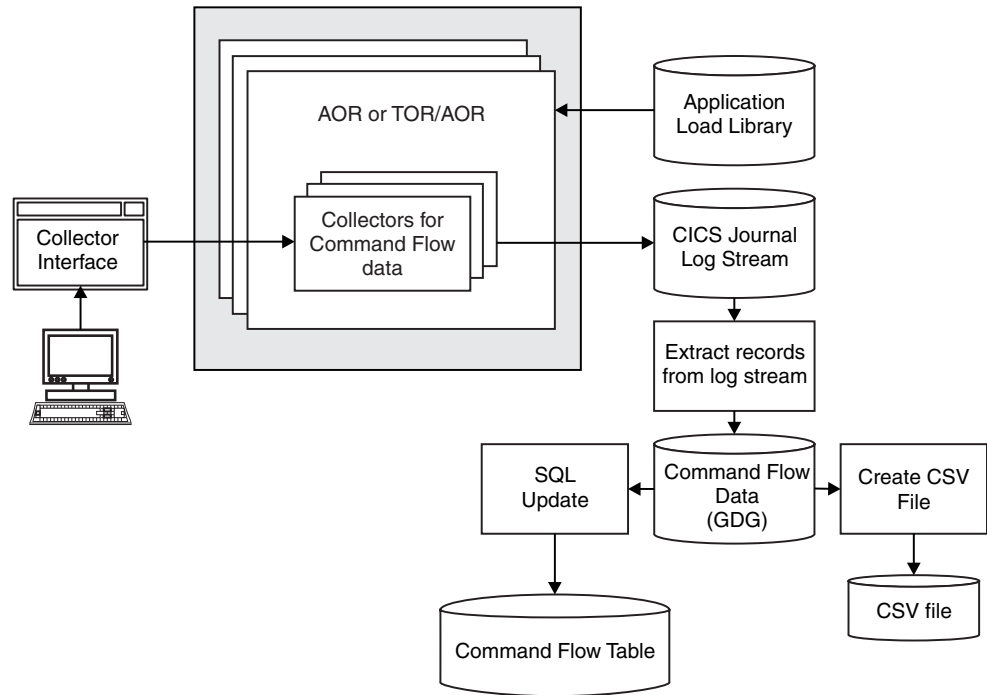


Figure 4. Command Flow option structure

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.

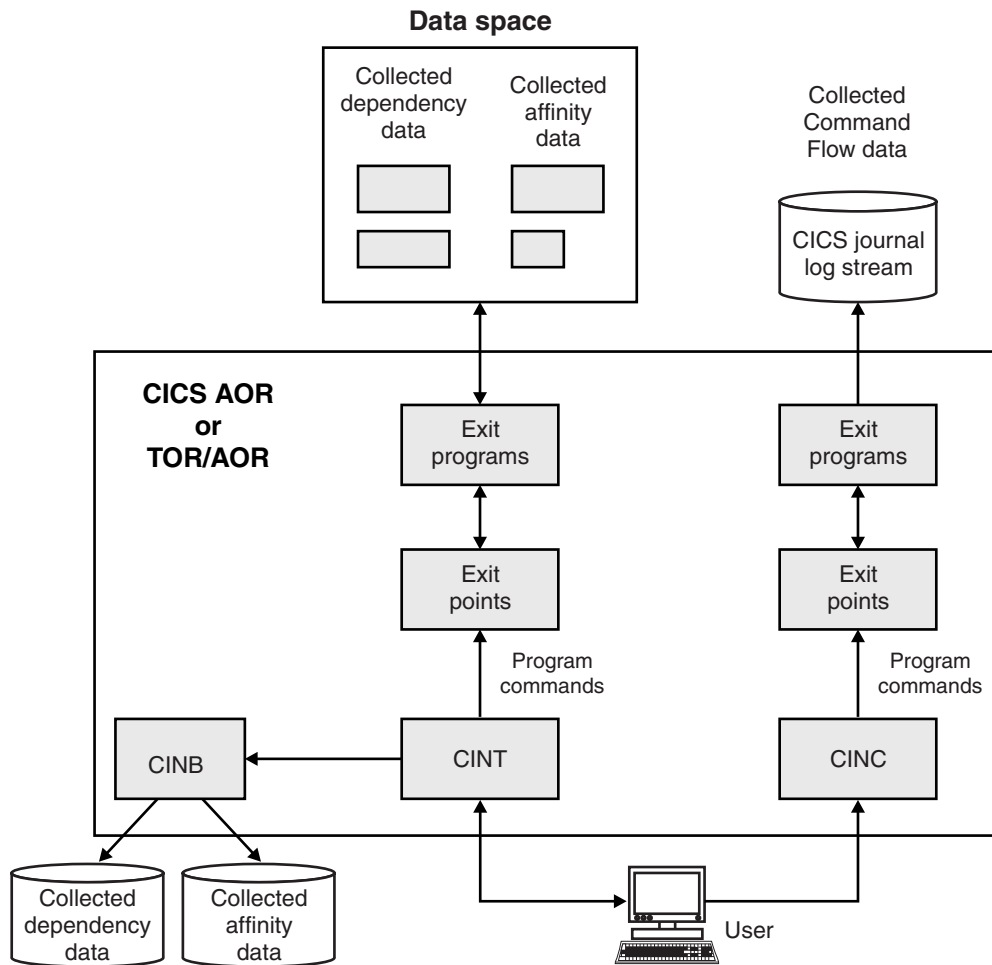


Figure 5. Collector components

Dependency data is collected by global user exit programs at the exit points. Affinity data is collected by the task-related and global user exit programs. Command Flow data is collected by the task-related and global user exit programs. 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 the Collector can monitor

The commands or calls that the Collector can monitor are listed. The commands or calls that the Collector actually monitors depend on how it is configured.

For more information about configuring the Collector, see “Controlling the Collector” on page 19 and “What the Collector does not monitor.”

The Collector can monitor the following commands or calls:

- The EXEC CICS API commands, listed in “CICS API commands” on page 217, that can cause transaction dependencies
- The EXEC CICS SPI commands, listed in “CICS SPI commands” on page 225, that can cause transaction dependencies
- The CICS Front End Programming Interface (FEPI) API commands, listed in “CICS FEPI API commands” on page 236, that can cause transaction dependencies
- The CICS FEPI SPI commands, listed in “CICS FEPI SPI commands” on page 236, that can cause transaction dependencies
- The non-CICS API commands, listed in “Non-CICS API commands detected” on page 237, that can cause transaction dependencies
- The EXEC CICS API and SPI commands, listed in Table 1 on page 13, that can cause transaction affinities
- The dynamic COBOL calls that are detected by the Dependency and Command Flow Data collectors for the IBM COBOL for OS/390 and IBM VS COBOL II programs if the CBLPSHPOP Language Environment option is active. The Language Environment option ALL31(ON/OFF) is supported for dynamic calls from IBM COBOL for OS/390 programs.

For more information about the CBLPSHPOP option, see Language Environment CBLPSHPOP option in the CICS TS 5.2 Knowledge Center.

For information about the detection of dynamic COBOL calls, see Appendix H, “Collecting dynamic COBOL calls,” on page 443.

For information about what the Collector reports for each monitored command, see Appendix A, “Details of dependencies and affinities collected,” on page 217.

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 the Collector does not monitor

The program commands or calls that the Collector does not monitor are listed.

The Collector does not monitor program commands in the following situations:

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

When it captures dynamic COBOL calls, 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.

For more information about the detection of dynamic COBOL calls, see Appendix H, “Collecting dynamic COBOL calls,” on page 443.

- The CICS IA Collector does not capture dynamic calls to PL/I and Assembler programs.
- The CICS IA Collector does not capture dynamic calls from PL/I and Assembler, including calls to COBOL programs.
- The CICS IA Collector correctly captures the EXEC CICS commands in the called programs.
- You are collecting interdependency data and the CICS command is not a command that is specified in the command exclude list. See “Creating a command exclude list” on page 76.
- 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 108.
- The command is not in the relevant subset of command types that is specified to be monitored. These subsets are as follows:
 - Dependency-related CICS commands. See “Specifying region-specific options: API and SPI commands to be monitored” on page 113.
 - Dependency-related CICSplex SM, DB2, IMS, and MQ commands. See “Specifying which dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored” on page 116.
 - Affinity-related CICS commands. See “Specifying which affinity-related CICS commands are to be monitored” on page 118.
- 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 74.

The default program exclude list excludes programs with names that start with ABL, CAU, CBM, CEE, CIU, CME, CPA, CSQ, DFH, DSN2, DSNCL, 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 75.
- The command causes an unhandledabend.

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.

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 94.
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 120.
- 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 108.
- 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 105.
- 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 105.
- 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 4, “Running the Collector,” on page 85.

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 and CICSplex SM table, which records every dependency on a CICS and CICSplex SM 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.

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 CICSplex SM, 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 and CICSplex SM 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 123.
- 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 105.
- 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 region-specific 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 135 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 251.

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 137 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 269.

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. When you install the CICS IA plug-in for CICS Explorer there is help available on how to use the CICS IA plug-in, **Help > Help Contents > CICS IA User Plug-in guide**.

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 information about configuring and using the CICS Explorer and the CICS Interdependency Analyzer plug-in, see <http://www-03.ibm.com/software/products/en/cics-explorer>.

This site guides you on how to download the CICS Explorer and the CICS IA plug-in for CICS Explorer. As part of the download you receive the `cicsia_plugin_release_notes.html` and the `cicts_explorer_release notes.html` files. These files provide you with guidance on how to install and configure the CICS IA plug-in.

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 or EXEC CPSM command that does not cause a TCB swap.

Non-Threadsafe

An EXEC CICS or EXEC CPSM 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 at 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 threadsafe from CICS TS V4.2 onwards.

MQ calls

The calls to the CICS MQ interface are threadsafe from CICS TS V3.2 onwards.

To request a 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 25 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 191.

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 the CICS IA plug-in for CICS Explorer

The CICS Interdependency Analyzer plug-in for CICS Explorer provides an Eclipse-based interface to analyze your CICS IA data. It also provides cheat sheets that guide you through the steps required to configure and install CICS IA. It is recommended that you download the CICS IA plug-in and follow the steps in the cheat sheets.

For information about configuring and using the CICS Explorer and the CICS Interdependency Analyzer plug-in, see <https://www.ibm.com/cics/explorer>. When you install the CICS IA plug-in for CICS Explorer there is help available on how to use the CICS IA plug-in, **Help > Help Contents > CICS IA User Plug-in guide**.

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

When you install the CICS IA plug-in for CICS Explorer there are cheat sheets supplied for configuring CICS IA, **Help > Cheat Sheets**. You can find additional help for using the cheat sheets by navigating to **Help > Help Contents > CICS IA User Plug-in guide**.

Note: The CICS IA plug-in connects to the CICS IA DB2 database. In order to use the CICS IA plug-in, you need to set up only this connection. You do not need to connect the base CICS Explorer to a CICS CPSM environment or a CICS single region with the CMCI connection.

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

Chapter 3. Configuring CICS IA

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

This section describes what you need to do before you can use CICS IA. You must follow the instructions in “Prepare your CICS IA configuration environment.” You can then chose the next parts depending on what you want to do.

-

Prepare your CICS IA configuration environment

CICS IA uses an ISPF dialog that runs REXX execs to assist you in configuring your CICS IA environments.

CICS IA takes the SMPE target data sets as input, updates the required members with the variable names you provide and then saves the information to customized output data sets. The customized data sets are then used to create your environments.

The ISPF dialog is split into three main categories:

- Configuring or updating your DB2 environment. CICS IA stores all the collected data in a CICS IA database. The CICS IA plug-in can then access the data through a JDBC Type 4 connection. Use the ISPF dialog to input the DB2 variables that are required, such as DB2 version. These variables are used to populate the output data sets for DB2 configuration. You can use the output data sets to run the tasks that are required to create the DB2 environment. These tasks include the following:
 - Creating the CICS IA database.
 - Creating the Stored Procedures.
 - Binding and granting access to the CICS IA plans.
- Configuring or updating your Collector environment. The CICS IA collector uses several CICS GLUEs and TRUEs to collect the data that you request, and stores the data in VSAM data sets and user logstreams. Use the ISPF dialog to input the variables that are required. These variables are used to populate the output data sets for a Collection configuration. You can use the output data sets to run the tasks that are required to set up the collector. These tasks include the following:
 - Allocating the VSAM files.
 - Allocating the resources that are required for Command Flow collection.
 - Allocating the CICS resource definitions.
 - Allocating the CICS file resource definitions.
 - Allocating web service resource definitions.
 - Creating and binding DB2 objects that are required by the Collector.
- Configuring or updating a DB2 UDB environment. 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.

The sample tasks help you to load the tables from the CSV files. Use the configuration exec to customize the sample to task to your site's standards and naming conventions.

The ISPF dialog saves your customization variables in two places.

- In your ISPF profile data set that is allocated to the ISPPROF file at logon. This data set is typically 'userid.ISPPROF'. In this data set, there is a member called CIUCPROF.
- In an ISPTABL data set that is concatenated to your ISTLIB file. By default this file is allocated as 'userid.ISPTABL'. The data set name that you use is stored in your ISPF profile. You can change this name as described in "Modify your settings" on page 34.

Data set allocation

The CICS IA configuration exec allocates data sets that save your configuration output. These data sets are allocated by the esoteric device group SYSDA.

Some of the sample jobs that are used to allocate files that are required by CICS IA also uses the SYSDA esoteric device group. You can change this value by updating the OUTPUT DEVICE TYPE value in your configuration.

Upgrade from an earlier release

If you are upgrading from CICS IA V3.1, V3.2, V5.1, or V5.2 it is best practice to take a copy of your ISPTABL data set before you run the ISPF dialog. The ISPF dialog automatically upgrades your ISPTABL data set by creating new members for the ISPF table format that is used by CICS IA V5.3.

If you are upgrading from CICS IA V3.1, the configurations can be stored only in a userid.ISPTABL data set. The old CIUCNTB table is upgraded and stored in the new, CIUCIC53 and CIUDB253, members in the 'userid.ISPTABL' data set. If you want to share this data set with colleagues you can copy it to a shared data set under a HLQ of your choice, and change the ISPTABL name as described in "Modify your settings" on page 34.

If you are upgrading from CICS IA V3.2 the configurations in the CIUCICST and CIUDB2T tables are upgraded and stored in the new, CIUCIC53 and CIUDB253, tables.

If you are upgrading from CICS IA V5.1 the configurations in the CIUCICT5 and CIUDB2T5 tables are upgraded and stored in the new, CIUCIC53 and CIUDB253, tables.

If you are upgrading from CICS IA V5.2 the configurations in the CIUCIC52, CIUDB252, and CIUDB52 tables are upgraded and stored in the new, CIUCIC53, CIUDB253, and CIUDB53 tables.

Running the configuration exec

To start the installation customization program, run member CIUCNFG1 of the SCIUEXEC library.

About this task

The program requires two parameters to be passed to it:

- The high-level qualifier of the CICS IA data sets &HLQ.
- 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 might start the installation customization program by entering the following at an ISPF command line:

```
tso ex 'CICSIA.V530.SCIUEXEC(CIUCNFG1)' 'CICSIA.V530 ENU'
```

The Welcome screen is displayed:

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

                                     More:  +
Press Enter to proceed, PF8 to scroll down, PF7 to scroll up or PF3 to
Exit.

                Welcome to the CICS IA Customization Function

This function will assist you in customizing your CICS IA sample jobs
and sample SQL.

Note: Before proceeding with this function please read chapter 3,
      "Preparing to use the CICS IA", in the "User's Guide and
      Reference". Also, please consult your DB2 Administrator.

All customized members will be copied from:

CICSIA.V530.SCIUSAMP
CICSIA.V530.SCIUSAME
CICSIA.V530.SCIUSQL
CICSIA.V530.SCIUDAT1
CICSIA.V530.SCIUDAT2
```

Enter to continue and the **Options** menu is displayed.

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

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

Please select how would you like to configure the CICS region or DB2:

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

Modify your settings

Before you configure your DB2 environment or the Collector environment, you must first update your settings.

Select option 0 for settings and the Settings panel is displayed.

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

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

Please specify settings:

CICS IA ISPTABL . . . . . CICSIA53.TEST.ISPTABL

JOB CARD HEADER:
//_CIUJOB_ JOB USER=&SYSUID,NOTIFY=&SYSUID,
//_          CLASS=A,MSGCLASS=Y,REGION=0M
//*
//*

DATASET NAME AND PROGRAM NAME FOR SORT:
SORT DATA SET NAME . . . . . SYS1.SORTLIB
SORT PROGRAM NAME . . . . . SORT
```

From this panel you can configure the following options.

Note: All of the values are saved in your ISPF profile data set.

CICS IA ISPTABL

This field defines the data set where the ISPF table for CICS IA is stored. The field is in the CIUCPROF ISPF profile member. The first time that you run the configuration exec an ISPTABL is allocated based on your user ID called 'userid.ISPTABL', which is the default. If you want to save this in a data set you can share with your colleagues, you can change this data set name to suit your requirements. When one of your colleagues uses the configuration EXEC, they must update their data set name to match. The name is then stored in their ISPF profile. If you are upgrading from a previous release of CICS IA, take a copy of the data set.

REQUIRED.

Note: Do not set this data set name to the SCIUTLIE/K data set sent with CICS IA.

JOB CARD HEADER

Enter a JOBCARD that is valid for your location.

REQUIRED.

SORT DATA SET NAME

The data set name for the SORT load library. The default is SYS1.SORTLIB

REQUIRED.

SORT PROGRAM NAME

The SORT program name. The default is SORT.

REQUIRED.

Short and Full configurations

You can select the Short or Full configuration option to create or change DB2, CICS, and UDB configurations.

If you select a Short configuration, you must specify site-dependent CICS and DB2 variables only. Other variables that relate to IA resources are set implicitly:

- For a new Short configuration, CICS IA implicitly sets default values for IA variables.
- For an existing Short configuration, CICS IA takes these IA variables from the existing configuration.

If you select a Full configuration, you must specify all of the required variables.

New DB2 configuration

The data collected from the Dependency, Affinity and Command Flow collectors are stored in the CICS IA database. Data collected from the CICS IA scanners are also stored in this database.

DB2 considerations

Before you run the installation jobs to create the DB2 environment, you must consider and decide the following questions.

- How many environments do I require and where do I create the environment?

Typically you need only one CICS IA database environment, and this environment is typically created in a development or test environment. All the data that is captured from the collector and scanners can be loaded into this single database. The data can include data that is collected from different environments, such as, development, test, and production. Storing the data in one place means that you can have a single connection to the CICS IA plug-in and the ability to compare resource usage across environments.

- Which versions of DB2 does CICS IA support?

DB2 versions and compatibility mode

CICS IA supports DB2 Version 8.1 and later.

If you are planning to install CICS IA with a DB2 Version that is running in compatibility mode, ensure that you set the DB2 Compatibility Mode option to YES when you run the configuration EXEC. This flag, and the DB2 version flag, is used by the configuration dialog to decide which resources you need to define.

DB2 Stored Procedures

CICS IA uses DB2 Stored Procedures to run 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 that are returned directly to an application, or access to data without granting the privileges to the applications.

CICS IA supports both Native and External SQL Stored Procedures. Native Stored Procedures are only supported in DB2 V9.1 or later. For more information about

available stored procedures, see Appendix G, “CICS IA External Interfaces,” on page 407. External DB2 Stored Procedures are run in a started task that is called WLM (Workload Manager) associated with each DB2 subsystem.

- How do I access the DB2 data in the IA database?

Access required for viewing the stored data

When you want to view data, the access is usually with the CICS IA plug-in for CICS Explorer. This access is through JDBC Type 4.

The batch jobs that are used to run reports against the CICS IA database use a combination of COBOL programs and DB2 stored procedures. These batch jobs are bound into a package and all members of that package list are bound into a plan. The name of the package and the plan are configurable. The users that need to run the reports require access to this plan. For more information about granting access to plans, see “Granting access to the plans and tables.”

Access required for loading the stored data

The batch jobs that are used to load data into the CICS IA database use a combination of COBOL programs and DB2 stored procedures. These batch jobs are bound into a package and all members of that package list are bound into a plan. The name of the package and the plan are configurable. The users that need to run the load utilities require access to this plan. In CICS IA V5.1 the same plan is used for both the report jobs and the load jobs. You can customize the supplied sample to package and bind these batch jobs into different plans and grant access to the plans as required.

Note: The batch jobs that are used to load all CICS IA database use the DB2 LOAD and UNLOAD utilities. The user ID or group that is used for these jobs requires DB2 Application Development Manager authority on the CICS IA database.

For more information about granting access to plans, see “Granting access to the plans and tables.”

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. There are two options.

About this task

Read this information with the CICS DB2 Guide.

Option 1

CICS IA uses the DYNAMICRULES(BIND) option on the BIND PLAN command in CIUDBNB. This option is recommended for the following reasons:

- How 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 of the security checks are done at the plan level. This option is simple to administer and offers good performance.
- If, typically, the _xxxx_ in the **GRANT EXECUTE** command specifies a RACF® group rather than a single RACF user ID, to add new users you connect the users to the RACF group.

The sample installation jobs CIUDBNB are configured to issue **GRANT EXECUTE** commands for the appropriate plans. These commands are issued against a RACF group.

1. Select your RACF group. The RACF group can be selected at configuration time.
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 installation job DSNTIJEX.

Note:

- a. Review DSNTIJEX job with your DB2 administrator.
- b. For a full understanding of the implications of DYNAMICRULES(BIND), see the description of the **BIND COMMAND** in the *DB2 Commands* manual.
- c. See also the section on DB2 security in the *CICS RACF Security Guide*.
- d. 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 must 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 evaluate the query to ensure that the existing indexing supports the query. If the existing indexing does not support the query, you must construct more indexes. Contact your database administrator if you require assistance.

Configure a new DB2 environment

To create a new DB2 configuration, select option 1 from the configuration menu.

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

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

Please select how would you like to configure the CICS region or DB2:

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

On the following panel, complete the fields and press Enter to continue.

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

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

Short or Full Configuration      S (S/F)

Please enter configuration name, DB2 version and d escription:

Configuration . . . . . DB2CONF1
DB2 version . . . . . V910
Description . . . . . DB2 Configuration
```

To configure CICS IA for a DB2 database, you must complete the following fields:

SHORT OR FULL CONFIGURATION

Set this option to S (Short), if you want to create a new collection (CICS) configuration by specifying the minimum number of required variables. All of the other variables are set to default values. Set this option to F (Full), if you want to specify all of the required variables.

REQUIRED.

CONFIGURATION NAME

The name of the upgraded DB2 configuration. Set this variable to something meaningful such as the name of the DB2 subsystem in which you are creating the CICS IA DB2 database.

REQUIRED.

DESCRIPTION

A short description of the DB2 configuration.

OPTIONAL.

Output data set variables

The values that are required to allocate the output data sets.

OUTPUT DSN FOR DB2 SCIUSAMP

The output data set that contains the modified SCIUSAMP members to configure the DB2 database. This field defaults to “@hlq.SCIUSAMP.DB2.”

REQUIRED.

OUTPUT DSN FOR DB2 SCIUSQL

The output data set that contains the modified SCIUSQL members to configure the DB2 database. This field defaults to “@hlq.SCIUSQL.DB2.”

REQUIRED.

OUTPUT DSN FOR DB2 SCIUDAT1

The output data set that contains the modified SCIUDAT1 members to configure the DB2 database. This field defaults to “@hlq.SCIUDAT1.DB2.”

REQUIRED.

OUTPUT DSN FOR DB2 SCIUDAT2

The output data set that contains the modified SCIUDAT2 members to configure the DB2 database. This field defaults to “@hlq.SCIUDAT2.DB2.”

REQUIRED.

OUTPUT DEVICE TYPE

The data set device type that is used to allocate the output data sets. This field defaults to SYSDA.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DATA SET VERIFICATION

Defines whether it is necessary to verify the existence of the data sets required to configure and create the CICS IA database. For example, the DB2 load library data set SDSNLOAD. This field defaults to YES.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

HLQ for CICS IA output data sets

This field is displayed on the panel for a Short configuration only. It is required and specifies the common value for both the DB2 LOAD TEMPLATE QUALIFIER and the QSAM FILE QUALIFIER variables. For more information, see “DB2 variables.”

DB2 variables

The DB2 variables that are required to define the CICS IA DB2 database

It is recommended that you consult with your DB2 systems programmer before you set these values.

DB2 VERSION

The DB2 version number. This variable is used with DB2 compatibility mode to decide on what is configured in the CICS IA database. This field defaults to V910.

REQUIRED.

DB2 COMPATABILITY MODE

DB2 sub systems can run in compatibility mode (CM) and in new function mode (NF). For example, DB2 V10 on z/OS in CM is designed so that DB2 V10 can be used as a drop in replacement for DB2 V9. If you are in compatibility mode, set this field to YES. This field is used with the DB2 version number to decide on what is configured in the CICS IA database.

REQUIRED.

DB2 LOAD DATA SET NAME

The data set name for the DB2 SDSNLOAD library.

REQUIRED.

DB2 RUNLIB DATA SET NAME

The data set name for the DB2 RUNLIB.LOAD library.

REQUIRED.

DB2 PROCLIB DATA SET NAME

The data set name for the DB2 PROCLIB library that contains the DSNUPROC.

REQUIRED.

DB2 LOAD TEMPLATE QUALIFIER

The data set qualifier that is used for allocating DB2 copy data sets when you use the DB2 LOAD utility. The qualifier is concatenated with the date and time stamp to create a unique data set name.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value that is specified for "HLQ for IA output data sets" .

DB2 SUB SYSTEM

The DB2 sub system identifier in which the CICS IA DB2 data sets are created.

REQUIRED.

DB2 DATABASE FOR TBLSPCS

The IA DB2 database name.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CIU53DB.

DB2 TABLE QUALIFIER

The DB2 qualifier, which is sometimes referred to as schema for the database objects. This value is used to set the SCHEMA during the creation of the DB2 objects and as the QUALIFIER when the CICS IA DB2 batch programs bind. This value is also required when you connect through the CICS IA plug-in.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CIUIA53.

DB2 TABLE OWNER

The TSO user ID that owns the DB2 resources. This value is used to set the

CURRENT SQLID during the creation of the DB2 objects and as the OWNER when the CICS IA DB2 batch programs bind.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to &SYSUID.

DB2 BUFFERPOOL FOR TBSPC

The DB2 BUFFERPOOL value that is used in the CREATE TABLESPACE DDL statements.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to BP1.

DB2 BUFFERPOOL FOR INDEX

The DB2 BUFFERPOOL value that is used in the CREATE INDEX DDL statements.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to BP1.

USE EXISTING STORAGE GROUP

If you want to use existing storage groups for your DB2 STOGROUP values, set this value to YES. If you want CICS IA to create new DB2 storage groups, set this value to NO. CICS IA uses the values that are provided for DB2 DASD VOLUME and DB2 VCAT QUALIFIER to create the storage groups. This field defaults to NO.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DB2 STOGROUP FOR TBLSPCS

The DB2 Storage group value that is used in the CREATE TABLESPACE DDL statements. You can use existing storage groups or request CICS IA to create new storage groups.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CIU53SPC.

DB2 STOGROUP FOR INDEXES

The DB2 Storage group value that is used in the CREATE INDEX DDL statements. You can use existing storage groups or request CICS IA to create new storage groups.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CIU53NDX.

DB2 VCAT QUALIFIER

The DB2 data set qualifier under which the CICS IA DB2 data sets are cataloged. This value is used when you create a new CICS IA DB2 storage groups.

REQUIRED if you want new storage groups to be created.

DB2 DASD VOLUME

The DASD volume on which DB2 resources are cataloged. This value is used when you create new CISC IA DB2 storage groups.

REQUIRED for a Full configuration, if new storage groups are required. For a Short configuration, this field is not displayed and defaults to SYSDA.

DB2 PLAN NAME FOR DSNTEP2

The DB2 plan name for the DB2 utility DSNTEP2.

REQUIRED.

DB2 PLAN NAME FOR DSNTIAUL

The DB2 plan name for the DB2 utility DSNTIAUL.

REQUIRED.

DB2 PLAN NAME FOR BATCH

CICS IA uses a number of COBOL DB2 programs in batch to load data into the CICS IA database. These programs are bound and packaged into this DB2 plan. Access to this plan can then be granted to the required RACF group or user ID.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to IA53BTCH.

DB2 COLLECTION ID

The DB2 collection identifier for the CICS IA DB2 packages. This value is used during the bind of the DB2 packages and the DB2 batch plan that CICS IA uses. It is also used when you define the WLM Stored procedures.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to IA53COLL.

DB2 AUTHORIZATION ID

The DB2 authorization ID that is used to grant access to the CICS IA DB2 plans. The authorization ID can be set to a RACF user ID, a RACF group or PUBLIC. CICS IA uses a number of COBOL DB2 programs to load data into the database. These programs are bound and packaged into a DB2 plan. Access to this plan is then granted to the authorization ID.

OPTIONAL. For a Short configuration, this field is not displayed.

DB2 WLM PROCEDURE NAME

CICS IA uses a number of DB2 stored procedures to process data. The CICS IA stored procedures are supplied in two formats.

If you are on DB2 V810 or earlier, CICS IA uses linked stored procedures. These types of stored procedures need to be added to a DB2 WLM address space. This field is used to specify the WLM address space that is used when you define the stored procedures to DB2.

For DB2 V910 and later, Native DB2 stored procedures are supplied. Native stored procedures do not require a WLM address space. For DB2 V910 and later, leave the field empty.

OPTIONAL.

DB2 CCSID (code page)

CICS IA uses the DB2 LOAD utility to load the collected data into the CICS IA DB2 database. Use the CCSID option of the LOAD utility statement to specify the CCSIDs of the data in the input file.

REQUIRED.

General variables

The variables required to allocate the QSAM data sets that are used by CICS IA batch jobs.

Depending on whether you configure a short or full configuration, you must specify the QSAM variables that are required to use the CICS IA batch utility.

QSAM DATA SET QUALIFIER

The qualifier prefix for CICS IA QSAM data sets.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for "HLQ for IA output data sets".

QSAM FILE SPACE UNITS

The space units to specify the data set allocation size that is required for the QSAM data sets.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CYLINDERS.

QSAM FILE PRIMARY QTY

The primary allocation quantity for the QSAM files, in cylinders, tracks, kilobytes, or megabytes as selected in the QSAM FILE SPACE UNITS field.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to 10.

QSAM FILE SECNDRY QTY

The secondary allocation quantity for the QSAM files, in cylinders, tracks, kilobytes, or megabytes as selected in the QSAM FILE SPACE UNITS field.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to 2.

Create a new DB2 environment

This section describes the steps that are required to define your CICS IA DB2 database, and how to review and submit the sample jobs that you configured in the previous section.

When you have run the configuration utility to define your DB2 environment variables you should have the following output data sets.

- hlq.SCIUSAMP.DB2
- hlq.SCIUSQL.DB2
- hlq.SCIUDAT1.DB2
- hlq.SCIUDAT2.DB2

The data set hlq.SCIUSAMP.DB2 contains the sample jobs that are required to define and maintain your CICS IA DB2 environment.

Step 1 - Creating the IA DB2 database

To create the CICS IA database, follow these steps:

1. Review sample job CIUDBCQ in hlq.SCIUSAMP.DB2.
2. Review the associated SQL member, CIUMAIN, in hlq.SCIUSQL.DB2.
3. Run sample job CIUDBCQ to create the database objects.

If you are creating the database on DB2 V9 or later, follow these steps to create the DB2 Stored procedures::

1. Review sample job CIUDBCP1 in hlq.SCIUSAMP.DB2.
2. Review the associated SQL member, CIUSPDL1, in hlq.SCIUSQL.DB2.
3. Run sample job CIUDBCP1 to create the database objects. This job should complete with return code 0.

If you are creating the database on DB2 V8 or earlier, follow these steps to create the DB2 Stored procedures: :

1. Review sample job CIUDBCP2 in hlq.SCIUSAMP.DB2.

2. Review the associated SQL member, CIUSPDL2, in hlq.SCIUSQL.DB2.
3. Run sample job CIUDBCP2 to create the database objects. This job should complete with return code 0.

Step 2 - Binding the IA DB2 packages

1. Review sample job CIUDBNB in hlq.SCIUSAMP.DB2.
2. Review the associated SQL member, CIUGRNTB, in hlq.SCIUSQL.DB2. This sample member includes sample SQL to grant a RACF group or user ID access to the CICS IA plans. It is used in the STEP 3 of the sample job.
3. Run sample job CIUDBNB to bind the DB2 batch programs that are used by CICS IA sample jobs and to grant access to the bound plan. This job completes with a return code of 0.

Step 3 - Loading static DB2 tables

CICS IA uses the following static tables.

CIU_VERSION

The CIU_VERSION table contains CICS IA Version and Service information and is used by the CICS IA plug-in for CICS Explorer.

- To load the version table, review and run job hlq.SCIUSAMP.DB2(CIUVERLD). This job completes with a return code of 0

CIU_TRANSLATORS

The CIU_TRANSLATORS table contains a one to one relationship between the IBM program component numbers for compilers, translators, and linkage editors and a description for each component.

- To load the translator table, review and run job hlq.SCIUSAMP.DB2(CIUTLOAD). This job completes with a return code of 0.

CIU_THREADSafe_CMD

The CIU_THREADSafe_CMD table contains information on whether an EXEC CICS command is threadsafe or not, for each version of CICS TS. It is used by the threadsafe report program and stored procedure to determine whether a command is threadsafe.

- To load the threadsafe table, review and run job hlq.SCIUSAMP.DB2(CIUTSLOD). This job completes with a return code of 0.

CIU_TRUEEXIT_INFO

The CIU_TRUEEXIT_INFO table contains a one to one relationship between the CICS TRUEs that are used in your environment and a description for each TRUE. This table is used by the CICS IA plug-in for CICS Explorer.

- To add your own TRUEs to the table, edit and save hlq.SCIUDAT2.DB2(CIUTRCD).
- To load the TRUE exit table, review and run job hlq.SCIUSAMP.DB2(CIUTRLOD). This job completes with a return code of 0.

Step 4 - DB2 stored procedures set up

CICS IA uses DB2 Stored Procedures to run complex DB2 tasks.

If you are running DB2 V910 or later the CICS IA uses Native DB2 stored procedures, as defined in “Step 1 - Creating the IA DB2 database” on page 43.

If you are running DB2 V810 or earlier, then CICS IA uses external stored procedures, as defined in “Step 1 - Creating the IA DB2 database” on page 43.

External DB2 stored procedures run in a started task that is called WLM (Workload Manager) associated with each DB2 subsystem. Sample JCL for a WLM started task can be found in hlq.SCIUSAMP.DB2(CIUSPTSK). If you already have a DB2 WLM task, you need to add the CICS IA load library hlq.SCIULOAD to the STEPLIB concatenation in this JCL.

Note: The name of the started task must match the one supplied in the CICS IA configuration variable DB2 WLM PROCEDURE NAME.

For more information about implementing DB2 Stored Procedures, see the *DB2 for z/OS Administration Guide*.

Loading the IVP data and connect to the CICS IA plug-in

CICS IA provides sample data that you can load into your new IA DB2 database. You can then connect to the database with the CICS IA plug-in for CICS Explorer and verify that your DB2 database is created successfully.

Loading the IVP sample data

- To load the IVP sample data, review, and run job hlq.SCIUSAMP.DB2(CIUIVPLD). This job completes with a return code 0 or 4.

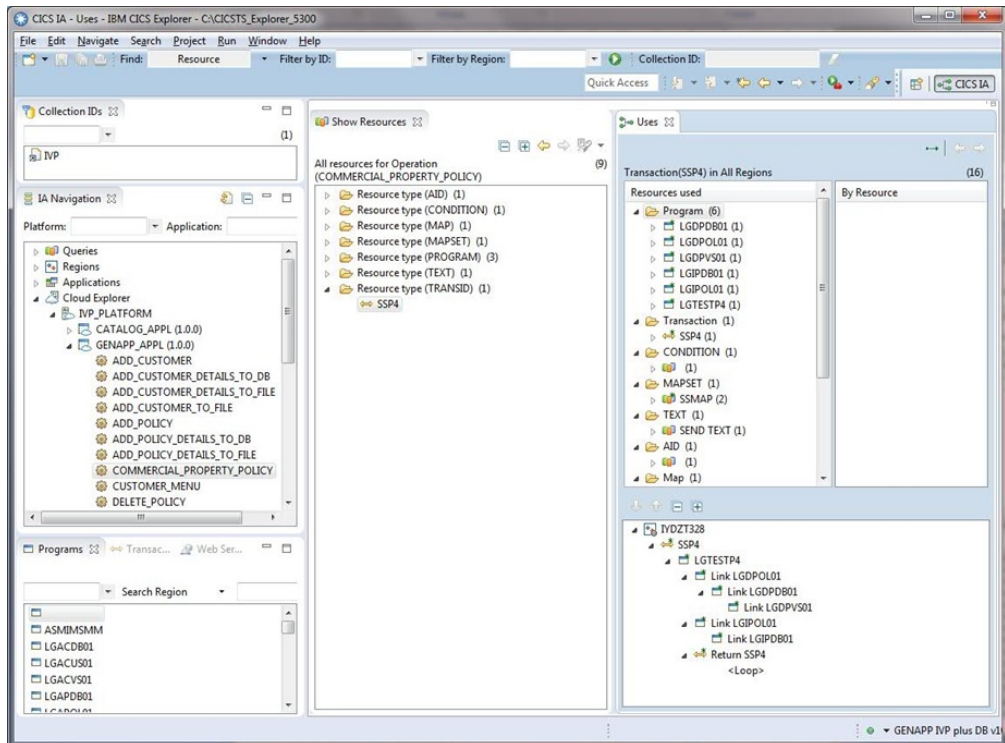
Viewing the IVP sample data

Before you complete this task, you must have the CICS IA plug-in installed into your CICS Explorer environment with an established a connection to the CICS IA perspective. For more information about connecting to the CICS IA perspective, see Chapter 2, “Getting Started with the CICS IA plug-in for CICS Explorer,” on page 29.

When you have a connection that is established and you have selected the CICS IA perspective, do the following tasks:

1. In the CICS IA plug-in, open the **IA Navigation** view.
2. Select **Window > Show View > Other**
3. Enter IA in the filter box and select the **IA Navigation** view.
4. Expand **Cloud Explorer**.
5. Expand **IVP_PLATFORM** and **GENAPP_APPL**.
6. Right-click the **COMMERCIAL_PROPERTY_POLICY** application and click **Show All Resources**. All resources for the selected operation are displayed in the **Show Resources** view.
7. In the Show Resources view, expand the **TRANSID** resource type.
8. Right-click the **SSP4** transaction, then click **Uses Resources > All Regions**. The results are displayed in the **Uses** view.

The following example screen capture shows the CICS IA perspective:



Your CICS IA database is now successfully configured and created.

New collection configuration

CICS IA provides a number of real-time collectors. To enable the collectors, you must complete a number of tasks to create the resources that are required to facilitate the collection and to define these resources to the CICS regions where the collectors run.

CICS considerations

Before you run the installation jobs that enable your CICS regions to collect CICS IA data, you must consider and resolve the following questions.

- In which CICS regions do I want to collect CICS IA data?
- In which types of CICS regions should I collect CICS IA data?
- In which one CICS region can I administer and operate CICS IA?
- Do I want to collect DB2 command in this CICS region?
- Do I want to use the Command Flow collector?

The CICS IA collector can be enabled in your development, test, QA, and production environments. The CICS IA collector can be enabled in different types of CICS regions that include the following:

- Application owning regions (AORs).
- Data Owning regions (DORs, FORs, TSQueue owning regions).
- Terminal Owning regions.

CICS IA provides three types of collectors:

- The dependency collector

- The affinity collector
- The command flow collector

The Dependency Collector captures data on the commands and resources that are used by a Transaction and Program in the CICS region. Application programs are typically run in AORs, so it makes sense to enable CICS IA in your AORs at the start.

If you want to answer questions such as:

- Which CICS regions use the FILE resource PAYFILE?
- Which DB2 tables are used by my CICS regions?
- Which TSQUEUES are used by my CICS regions?

You can enable CICS IA in your Data Owning regions.

The affinity collector captures information that you can use to proceed with deploying CPSM Workload Management. The affinity collector helps identifying transactions that must remain in the same CICS regions as it started because it has an affinity to that region. If you are planning to clone an application region and use CPSM WLM to manage the work across these regions, you must enable CICS IA in this region.

The command flow collector captures information about all the commands that are issued by your CICS tasks, in chronological order. The command flow collector can capture information from the start of a transaction in a TOR, the commands that are issued by your application in the AOR. When you activate the command flow collector you can choose which regions to start the collector in, you must enable CICS IA in all of those regions.

If you are using cloned regions, you might want to consider enabling CICS IA in one or two of these regions, but not all your cloned regions. The data that you collect ought to be the same. You might think that these regions are exact clones, however, by collecting CICS IA data in two different clones you can view and compare which resources and commands are being used.

CICS IA controlling region

CICS IA stores all the information about which collections and what data you want to collect, and so on, in a VSAM file. This file is known as the CONTROL file.

The IA Dependency and Affinity Collectors save the data collected into VSAM files. These files are called COLLECTOR files. The command flow collector saves the data into CICS User journals, (logstreams).

CICS IA is designed to share these VSAM files and logstreams across CICS regions, so you can administer and operate CICS IA from one region. This region is called the CICS IA controlling region. From this controlling region you can administer the CICS IA options and operate, start and stop, the CICS IA collectors. To operate the CICS IA collector the controlling region needs a connection to the regions that you want it to start or stop.

Share 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 and affinity data files, and the control record file across all the regions that are monitored.

To share the files, you can use either:

- VSAM record-level sharing (RLS). For more information about using VSAM RLS in CICS, see the *CICS Installation Guide*.
- Function shipping to a file-owning region (FOR). For more 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 controlling region or in a file owning region and remote in all the other regions to be monitored.

You can select your preferred sharing method and customize the required samples according to your selection, with the customization exec.

Controlling CICS IA from the CICS IA plug-in

You can administer and operate the CICS IA collectors from the CICS IA plug-in for Explorer.

You can stop or start the Dependency, Affinity, and Command Flow collectors from the CICS IA plug-in. You can also administer the Command Flow collector options from the CICS IA plug-in. CICS IA uses a web service resource to enable this connection to the CICS IA plug-in. Web service resources were introduced in CICS TS V3.1. If you want to use this feature, then you must install web service resources into the controlling region.

You can choose if you want to configure the sample jobs that are required to enable this feature by using the customization EXEC, started in full mode. To use the sample jobs, in the configuration EXEC, select YES for **CICS IA Control Region**. A TCP/IP port number must be associated with the web service. The zFS directory values that are associated with the web service definition default to the zFS directory values used at the SMPE installation time for CICS IA and CICS TS.

For more information, see “CICS Explorer plug-in variables” on page 56 and “Creating a new collection environment” on page 59.

For more information about how to configure the CICS IA Explorer plug-in, see the CICS IA plug-in help.

Web service security: The TCPIPService resource definition contains the AUTHENTICATE(BASIC) parameter. HTTP Basic authentication is used to obtain a user ID and password from the plug-in. Basic authentication requires the SEC system initialization parameter to be set to YES. If SEC is set to NO, then the TCPIPService resource with BASIC authentication cannot be installed in the region. If you do not want to use this type of authentication, define AUTHENTICATE(NO) in the CIUDEFW sample file before the CIUJCWEB job is run. However, if you do not use this authentication, you can work from the plug-in only as the CICS controlling region default user.

Collecting DB2 commands

You can collect data about the DB2 commands and resources that are used by your transaction and program in a CICS region with the CICS IA dependency collector.

To obtain the resource name, for example a table name that is used in an **EXEC SQL SELECT** command, you need to access the SYSIBM.SYSPACKSTMT and the SYSIBM.SYSSTMT tables. To improve the performance of this access, CICS IA supplies you with some DB2 indexes for these tables. It is recommended that you create these indexes. Consult with your DB2 Administrator. Some locations are reluctant to create indexes on these tables. In this case it is recommended that you

use the CICS IA option, Collect resource name, to remove access to these tables. For more information, see “Specifying which dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored” on page 116.

To enable the collection of DB2 data, you must provide some DB2 options to the configuration exec. Consult with your DB2 Administrator to obtain these values. If you choose to enable DB2 collection in your CICS region, CICS IA defines DB2ENTRY and DB2TRAN resource definitions. A DB2CONN definition is not supplied.

Note: The DB2 subsystem that is connected to your CICS system and accessed by your applications can be different from the DB2 subsystem where you created the CICS IA database.

Command flow collector

The CICS IA command flow collector uses CICS user journals or logstreams, to store captured data. The logstreams can be created as coupling facility logstreams or DASD only logstreams.

The CICS IA command flow collector can be used in single-user mode or multi-user mode. The single-user mode was designed for systems programmers to capture the command flow of a transaction in chronological order and to view the TCB modes that are used by each command to assist with threadsafe analysis. In CICS IA V5.1 support was introduced for multiple users. To support this feature, the logstream must be shared by multiple users. When you create the logstream, you must review four values:

AUTODELETE

If you want to give access to multiple users, set this value to YES.

If you want to restrict the usage to a single user, set this value to NO.

This value can be set in the configuration exec.

RETPD

This value is the retention period, in days, for the data to be written into the logstream.

For multiple users set the value to the number of days you want users to keep data in the logstream before it is offloaded to the CICS IA GDG data sets.

For a single user, you can set this value to 0. For single user usage CICS IA supplies a sample job, found in SCIUSAMP.CICS(CIUJLDEL) so that you can delete data from the logstream when required.

This value can be set in the configuration exec.

HIGHOFFLOAD and LOWOFFLOAD

Data from a logstream can be offloaded to DASD data sets when the logstream use, in the coupling facility or the staging data set, reaches the HIGHOFFLOAD limit. The amount of data that is offloaded is determined by the LOWOFFLOAD limit.

The recommendations for user journals are different from the system log. There is no requirement to retain logged data in the coupling facility structure. Due to the typical use of such data, you might need only a small structure and offload the data rapidly to DASD.

If this is your requirement, default HIGHOFFLOAD to 80 and LOWOFFLOAD to 0. CICS IA uses these values as defaults.

If you want to define the logstream in a coupling facility, you must provide a CF STRUCTURE name. You can create a structure specifically for CICS IA or use a general one that is already defined. You can use a CF structure that is created for User Journals. For example, LOG_USERJRNL_001.

The data in the logstream is offloaded into the CICS IA generation data group files. If you are using the Command Flow collector in multi-user mode then, it is recommended that you create a GDG definition job for each user. CICS IA sample job SCIUSAMP.CICS(CIUJCLCG) is provided to create the GDG files for each user. The sample job SCIUSAMP.CICS(CIUJLCPY) is provided to offload from the shared logstream to the individual users GDG data sets.

Security considerations

For information about how to set up RACF security for CICS IA, see Appendix F, “CICS IA security,” on page 405.

Configuring a new collection environment

To create a new collection configuration, select option 3 from the configuration menu.

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

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

Please select how would you like to configure the CICS region or DB2:

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

On the following panel, complete the fields and press Enter to continue.

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

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

Short or Full Configuration . . . S  (S/F)

Please enter CICS configuration name and description:

CICS Configuration . . . CICSCONF
Description . . . . . Collection configuration
```

To configure the CICS IA collector, you must complete the following fields:

CICS configuration

SHORT OR FULL CONFIGURATION

Set this option to S (Short), if you want to create a new collection (CICS) configuration by specifying the minimum number of required variables. All of the other variables are set to default values.

Set this option to F (Full), if you want to specify all of the required variables.

REQUIRED.

CONFIGURATION NAME

The name of the new CICS configuration. Set the name to something meaningful such as the name of the CICS job , the CICS APPLID or a CPSM CICS group definition.

REQUIRED.

DESCRIPTION

A short description of the CICS configuration.

OPTIONAL.

DB2 configuration

DB2 CONFIGURATION FOR IA

The name of the DB2 configuration that is used to create the DB2 database where the collected data is stored. You can enter the name of an existing DB2 configuration or press PF4 to select an existing one from the list.

Note: If you are configuring the collector to use with a UDB database, use "UDB" as the DB2 configuration name.

REQUIRED.

Output data set variables

The values that are required to allocate the configured output data sets.

OUTPUT DSN FOR CICS SCIUSAMP

The output data set that contains the modified SCIUSAMP members that are used to configure the CICS IA collector. This field is defaults to "@hlq.SCIUSAMP.CICS".

REQUIRED.

OUTPUT DSN FOR CICS SCIUSAME/K

The output data set that contains the modified SCIUSAME/K NLS members that are used to configure the CICS IA collector. This field is defaults to "@hlq.SCIUSAME/K.CICS".

REQUIRED.

OUTPUT DSN FOR CICS SCIUSQL

The output data set that contains the modified SCIUSQL members to configure the CICS IA Collector.

Note: If you are configuring the collector to use with a UDB database, this data set is not allocated.

This field is defaults to "@hlq.SCIUSQL.CICS". REQUIRED.

OUTPUT DSN FOR CICS SCIUDAT1

The output data set that contains the modified SCIUDAT1 members that are used to configure the CICS IA collector. This field is defaults to "@hlq.SCIUDAT1.CICS".

Note: If you are configuring the collector to use with a UDB database, this data set is not allocated.

REQUIRED.

OUTPUT DEVICE TYPE

The data set device type that is used to allocate the output data sets. This field is defaults to SYSDA.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DATA SET VERIFICATION

Defines whether it is necessary to verify the existence of the data sets that are required to configure and create the IA database. For example, the CICS load library data set SDFHLOAD. This field is defaults to YES.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

HLQ for IA output data sets

This field is displayed on the panel for Short configuration only. It is required and specifies the common value for IA VSAM FILE QUALIFIER, LOGSTREAM QUALIFIER and GDG DATASET QUALIFIER variables. These variables are described in the “CICS IA variables” and “CICS IA command flow variables” on page 53 topics.

CICS IA variables

The variables that are required to configure the logstream and GDG data sets that are required by CICS IA.

IA PRODUCT QUALIFIER

The CICS IA data set high-level qualifier. This variable is passed in as input when you start the exec. For a Short configuration, this field is not displayed.

IA VSAM FILE QUALIFIER

The qualifier prefix for the CICS IA VSAM data sets that are used by CICS IA collector.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for “HLQ for IA output data sets”.

IA VSAM FILE DATACLASS

The SMS data class that is used for the allocation of the VSAM data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA VSAM FILE STORAGECLS

The SMS storage class that is used for the allocation of the VSAM data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA VSAM FILE MNGMNTCLASS

The SMS management class that is used for the allocation of the VSAM data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA VSAM FILE SPACE UNITS .

The space units to express the data set allocation size that is required for the VSAM data sets.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CYLINDERS.

IA VSAM FILE PRIMARY QTY

The primary allocation quantity for the VSAM files, in cylinders, tracks, KB or megabytes, as selected in the **SPACE UNITS** field. This field defaults to 20.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

IA VSAM FILE SECNDRY QTY

The secondary allocation quantity for the VSAM files, in cylinders, tracks, KB, or megabytes, as selected in the **SPACE UNITS** field. This field defaults to 4.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DUMP QUALIFIER

This variable is the high-level qualifier for the dynamic memory dump data set at your location. This field defaults to DUMP.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

CICS IA command flow variables

The variables that are required to configure the Logstream and GDG data sets that are required by CICS IA.

COUPLING FACILITY or DASD

You can choose to create your user logstream in a coupling facility or on DASD-only. This field defaults to CF.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LOGSTREAM QUALIFIER

The qualifier prefix for the CIUMTJNL logstream data sets that are used by the Command Flow Collector.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value that is specified for "HLQ for IA output data sets".

LOGSTREAM DATA SET NAME HLQ.

The logstream HLQ, which is used for the logstreams. You can use the CICS default qualifier IXGLOGR. Ensure that an alias exists for the selected qualifier. This field defaults to IXGLOGR.

REQUIRED for a Full configuration if you are defining a DASD logstream. For a Short configuration, this field is not displayed.

LOGSTREAM HIGHOFFLOAD

The high offload limit when data in the logstream is offloaded to a data set. This field defaults to 80.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LOGSTREAM LOWOFFLOAD

The low offload limit that is used when data in the logstream is offloaded to a data set. This field defaults to 0.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LOGSTREAM RETENTION PERIOD

The retention period, in days, for data that is written to the logstream. This value is used with the AUTODELETE option when you create the logstream. The value can be in the range 0 - 999 days. This field defaults to 5.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LOGSTREAM AUTODELETE

If this field is set to YES, the data is automatically deleted from the logstream after a period determined by the LOGSTREAM RETENTION PERIOD. This field defaults to YES.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LOGSTREAM STRUCTNAME

The name of the coupling facility structure name where the logstream is defined. This field defaults to LOG_USERJRNL_001.

REQUIRED if you are defining a CF logstream. For a Short configuration, this field is not displayed.

GDG DATA SET QUALIFIER

The qualifier prefix for the CICS IA GDG data sets that are used by the CICS IA command flow collector.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value that is specified for "HLQ for IA output data sets".

NUMBER OF GDG DATASETS

Specifies the maximum number, 1 - 255, of GDSs that can be associated with the GDG being defined. This field defaults to 3.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

GDG DELETE OPTION

Specifies what action is taken for a generation data set when the data set is uncataloged from the GDG base as a result of NOEMPTY processing. If this field is set to SCRATCH, the oldest data set is physically deleted and uncataloged. If this field is set to NOSCRATCH, the data set is uncataloged but not physically deleted. This field defaults to SCRATCH.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

CICS variables

The CICS variables that are required to define CICS IA to the correct CSD locations.

CICS IA requires a number of resources to be defined to CICS. You configure where and how these resources are defined.

CICS VERSION

The CICS TS version for the CICS regions that you are configuring for the IA collection. This field defaults to V530.

REQUIRED.

CICS QUALIFIER

The qualifier prefix for the CICS TS SDFHLOAD library.

REQUIRED.

CICS CSD FILE NAME

The dataset name of the CICS system definition file (DFHCSD) where the CICS IA resources are added.

REQUIRED.

CICS AOR LIST NAME

CICS IA resources are defined in a number of groups. Select an appropriate list name used by your CICS AOR regions to which the IA groups are appended. This list must be included in the GRPLIST statement of the CICS SIT options.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to IA53AOR.

CICS CSD COLLECTOR GROUP

The group name for CINT resource definitions that are used by the CICS IA collector. This group is appended to the CICS AOR LIST NAME.

This field defaults to IA53CINT.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

CICS LOCAL FILES GROUP

The group name for the LOCAL FILE definitions. If you choose to share the VSAM files that use an FOR then the group is appended to the CICS FOR LIST NAME when you run job CIUJCFIL. If you choose to not to share your VSAM files or share using VSAM RLS then this group is appended to the CICS AOR LIST NAME when you run job CIUJCFIL.

This field defaults to IA53FILE.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

RLSACCESS FOR VSAM FILES

Set this value to YES if you choose to share the CICS IA VSAM files that use VSAM RLS. If not, set the field to NO.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to YES.

CICS FILE OWNING REGION

The CICS SYSID for the file owning region. This value is REQUIRED if you use a FOR to share VSAM files.

OPTIONAL. For a Short configuration, this field is not displayed.

CICS FOR LIST NAME

The CICS CSD list name for your File Owning Region. This field is REQUIRED if you specified NO for RLSACCESS FOR VSAM FILES and a nonblank value for the CICS FILE OWNING REGION. This list must be included in the GRPLIST statement of the CICS SIT options.

OPTIONAL. For a Short configuration, this field is not displayed.

CICS REMOTE FILES GROUP

The group name for the REMOTE FILE definitions. This field is REQUIRED if you specified NO for RLSACCESS FOR VSAM FILES and a nonblank value for the CICS FILE OWNING REGION. The group is appended to the CICS AOR LIST NAME when you run job CIUJCFIR.

OPTIONAL. For a Short configuration, this field is not displayed.

LSRPOOLID FOR VSAM FILES

If your CICS region uses LSRPOOLIDS select an appropriate value for your site. If not, set the field to N0.

This field defaults to NONE.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

LIBRARY RESOURCE NAME for SCIULOAD

CICS IA can be configured to use CICS LIBRARY definitions to define the CICS IA load libraries. Select a name for this resource for the base CICS IA library SCIULOAD. This field defaults to IA53LIB.

Note: NOTE: Your CICS region must be at version CICS TS 3.2 or later.

OPTIONAL. For a Short configuration, this field is not displayed.

LIBRARY RESOURCE NAME for SCIULODE/K

CICS IA can be configured to use CICS LIBRARY definitions to define the CICS IA load libraries. Select a name for this resource for the NLS CICS IA library SCIULODE/K. This field defaults to IA53LIBE.

Note: Your CICS region must be at version CICS TS 3.2 or later.

OPTIONAL. For a Short configuration, this field is not displayed.

CICS Explorer plug-in variables

The variables that are required to configure and install the web service resources that are required to connect to the CICS IA plug-in.

You can connect to the collector from the CICS IA plug-in with CICS IA by using the web service interface. The following fields are used to configure the location of the web service files and the TCP/IP port number that is used for the web service call.

CICS IA CONTROL REGION

If you want to configure the CICS resources that are required to connect the CICS IA plug-in to a controlling region, set this field to YES.

REQUIRED.

TCP IP PORT

The TCP/IP port number that is used in the web service call. The configuration EXEC uses the specified TCP/IP port number in the *Portnumber* field in the CSD TCPIP SERVICE resource definition. This definition enables the web service listener on the port.

REQUIRED if this region is used as a CICS IA controlling region.

WEB SERVICES FILES LOCATION

The zFS prefix path to the CICS web service binding directory, or pickup directory. This path is decided at SMPE installation time. View the installation data set member CIUMKDIR and look for the value of the *idir* variable. The suffix, /IBM/webservices/wsbind/provider is appended to the value specified in this field to build the full path to the pickup directory. The full path name is used by the configuration EXEC in the *Wsdir* field for the CSD PIPELINE resource definition.

REQUIRED if this region is used as a CICS IA controlling region. For a Short configuration, this field is not displayed and defaults to /usr/lpp/cicsia/V530.

WEB SERVICE SHELF

The zFS prefix path to the CICS web service *Shelf* directory. The suffix, /shelf is appended to the value specified in this field to build the full path to the Shelf directory. The full path name is used by the configuration EXEC in the *Shelf* field for the CSD PIPELINE resource definition.

REQUIRED if this region is used as a CICS IA controlling region. For a Short configuration, this field is not displayed and defaults to /usr/lpp/cicsia/V530.

CONFIGURATION FILE

The zFS prefix path to the CICS web service pipeline configuration file. This path is decided at CICS Transaction Server SMPE installation time. View the installation data set member DFHMKDIR and look for the value of the *idir* variable. The pipeline configuration file is an XML file, which includes descriptions of SOAP header processing and message handler programs. The suffix /samples/pipelines/basicsoap11provider.xml is appended to the value specified in this field to build the full path to the pipeline configuration file. The full path name is used by the configuration EXEC in the *COnfigfile* field for the CSD PIPELINE resource definition.

REQUIRED if this region is used as a CICS IA controlling region. For a Short configuration, this field is not displayed and defaults to /usr/lpp/cicsts/V530.

CSD WEB SERVICE CSD GROUP

This field defines the name of the web services resources group, which is created in the CSD file of the controlling region. The group contains the TCPIPSERVICE, PIPELINE, and PROGRAM definitions that are required to enable web service operations in CICS TS. This group is appended to the CICS AOR LIST NAME.

REQUIRED if this region is used as a CICS IA controlling region. For a Short configuration, this field is not displayed and defaults to IA53WEBS.

General variables

The variables that are required to enable CICS IA compilations of Assembler samples.

ASSEMBLER PROGRAM

The name of the assembler compiler for your location. The values can be ASMA90 or IEV90.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to ASMA90.

LE QUALIFIER

The qualifier prefix for the LE SCEERUN library.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CEE.

DB2 variables for CINT

The CICS IA collector can capture DB2 EXEC SQL commands that are issued by your applications. If the applications that run in the CICS regions you are configuring for do not use DB2, you can set the **COLLECTING DB2 FIELD** field to NO and complete the configuration.

Note: The DB2 subsystem that your CICS regions are connected to can be different to the DB2 sub system where the CICS IA database is defined.

COLLECTING DB2 INFORMATION

Set this field to YES or NO depending on whether you want to capture DB2 EXEC SQL commands.

REQUIRED.

DB2 VERSION

The DB2 version number. This variable is used with DB2 compatibility mode to decide which DB2 programs to bind and which DB2 resources are defined to enable the collection of EXEC SQL commands. This field defaults to V910.

REQUIRED.

DB2 COMPATABILITY MODE

DB2 sub systems can run in compatibility mode, CM, and in new function mode, NF. For example, DB2 V10 on z/OS in CM is designed for DB2 V10 to be used as a drop in replacement for DB2 V9. If you are in compatibility mode, set this field to YES. This field is used with the DB2 version number to decide which DB2 programs to bind and which DB2 resources are defined to enable the collection of EXEC SQL commands.

REQUIRED.

DB2 LOAD DATA SET NAME

The data set name for the DB2 SDSNLOAD library.

REQUIRED.

DB2 RUNLIB DATA SET NAME

The data set name for the DB2 RUNLIB.LOAD library.

REQUIRED.

DB2 SUB SYSTEM

The DB2 sub system identifier to which your CICS regions are connected.

REQUIRED.

DB2 TABLE QUALIFIER

The DB2 qualifier, sometimes referred to as schema, for the SYSIBM database tables that are accessed by the CICS IA collector. This value is used to set the QUALIFIER when you bind the CICS IA DB2 collector programs. This field defaults to SYSIBM.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DB2 TABLE OWNER

The TSO user ID that owns the DB2 resources that are required to enable the collection of EXEC SQL commands. This value is used to set the *CURRENT SQLID* during the creation of the DB2 objects and as the OWNER when you bind the CICS IA DB2 CICS programs.

REQUIRED.

DB2 BUFFERPOOL FOR INDEX

The DB2 BUFFERPOOL value that is used in the CREATE INDEX DDL statements. The CICS IA collector accesses SYSIBM tables to retrieve data about the EXEC SQL commands. CICS IA supplies sample INDEXES to improve the performance access to these tables.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to BP1.

DB2 PLAN NAME FOR DSNTEP2

The DB2 plan name for DB2 utility DSNTEP2.

REQUIRED.

DB2 PLAN NAME FOR CICS

CICS IA uses a number of DB2 programs in CICS to access SYSIBM tables to retrieve data about the EXEC SQL commands. These programs are bound and packaged into this DB2 plan. Access to this plan can be granted to the required RACF group or user ID. Access is required for only the user ID that the CICS IA collector is started by. This variable is typically the default CICS user ID. The plan name is also used to define the DB2TRAN and DB2ENTRY to CICS. This field defaults to IA53CICS.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DB2 COLLECTION ID

The DB2 collection identifier for the CICS IA DB2 packages. This value is used during the bind of the DB2 packages and the DB2 CICS plan that is used by CICS IA. This field defaults to IA53COLL.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

DB2 AUTHORIZATION ID

The DB2 authorization ID that is used to grant access to the CICS IA DB2 plans. The authorization ID can be set to a RACF user ID, a RACF group, or PUBLIC. CICS IA uses a CICS program to access SYSIBM tables to retrieve the data about the EXEC SQL command. These programs are bound and packaged into a DB2 plan. Access to this plan is then granted to the authorization ID. Set this field to the user ID under which CICS IA runs.

OPTIONAL. For a Short configuration, this field is not displayed.

Creating a new collection environment

The steps that are required to define the resources that are required to enable CICS IA collection in your CICS regions. Review and submit the sample jobs that are configured.

When the configuration utility runs to define your CICS environment variables, the following output data sets are available.

- hlq.SCIUSAMP.CICS
- hlq.SCIUSAME.CICS
- hlq.SCIUSQL.CICS
- hlq.SCIUDAT1.CICS

Note: If the CICS IA collector is configured to be used with a UDB database, SCIUSQL and SCIUDAT1 are not created.

The data set hlq.SCIUSAMP.CICS contains the sample jobs that you require to define and maintain your CICS IA collector environment.

Step 1 - Defining the VSAM files

To create the CICS IA VSAM files that are used by CICS IA:

To define the CICS IA Control file.

Review and submit the sample job CIUJCLCC in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

To define the CICS IA collector files.

Review and submit the sample job CIUJCLCA in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

Step 2 - Defining the Command Flow resources

The CICS IA command flow feature uses CICS logstreams and GDGs to facilitate the offloading of captured data into the DB2 tables. To create these resources:

To define the logstream that is used by CICS IA.

Review and submit the sample job CIUJCDLS in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

To define the user GDG data sets that are used by CICS IA.

Review and submit the sample job CIUJCLCG in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

Note: You must define GDG data sets for every CICS user ID that wants to use the command flow feature.

Step 3 - Defining the IA resources to CICS

Before defining your resources to CICS make sure that the GRPLIST you have selected is defined in your GRPLIST SIT parameter for the required CICS region.

To define the main CICS resources that are required by CICS IA.

Review and submit the sample job CIUJCINT in hlq.SCIUSAMP.CICS. This job is configured to include all the resources that are required to be defined, based on your selections, when you run the configuration exec. For example, if you chose to use LIBRARY names. These definitions are included. This job also defines the resources that are required for NLS support. The CSD and group names that are selected in the configuration exec are used. This job completes with a return code of 4 which is acceptable.

To define the VSAM file resources that are required by CICS IA.

If you are using a single CICS region or VSAM RLS to share the files, review and submit the sample job CIUJCFIL in hlq.SCIUSAMP.CICS. If you are using an FOR to share the files, then review and submit the following jobs:

- hlq.SCIUSAMP.CICS(CIUJCFIL) to define the CICS file resources locally or in a file-owning region. This job completes with a return code of 0.
- hlq.SCIUSAMP.CICS(CIUJCFIR) to define the CICS file resources as remote to a file-owning region. This job completes with a return code of 0.

To define the web service resources that are required by CICS IA.

Review and submit the job CIUJCWEB in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

Step 4 - Defining the DB2 resources

If you want to collect DB2 resource information in your target CICS region, you must follow these steps:

To define the DB2 indexes for the SYSIBM tables

Review and submit the sample job CIUDBCT in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

To bind and package the DB2 plans that are required for DB2 collection

Review and submit the sample job CIUDBNT in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

Note:

- If you select to configure CICS IA to collect DB2 resources, the DB2 resources for DB2TRAN and DB2ENTRY are defined in “Step 4 - Defining the DB2 resources.”
- On any region where you want 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 SCIUSAMP.CICS(CIUDBNT). This permission enables the background transaction CINB to access the SYSIBM.SYSDUMMY1, SYSIBM.SYSPACKSTMT, and SYSIBM.SYSSTMT DB2 tables.

Step 5 - Tailoring your CICS startup job

To enable CICS IA to run in your CICS region, tailor your CICS startup JCL as follows:

1. Set the ICVR system initialization parameter to a minimum of 10 seconds; that is, ICVR=10000 or a larger value. If you do not set 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. If you did not select to use the LIBRARY definitions or the CICS region is CICS TS V3.2 or earlier, you must add the following load libraries to the DFHRPL concatenation in the startup job JCL:
 - hlq.SCIULOAD.
 - hlq.SCIULODE (the default, English, national language: always required).
 - Add hlq.SCIULODJ (if you require Japanese national language support)
4. Add the following DD statement for the CINT transient data message log:
//CIULOG DD SYSOUT=*
5. If this is a CICS IA Controlling Region and you are installing the web services resources, you require the following SIT options:
 - TCPIP=YES
 - SEC=YES

The TCPIP SERVICE resource definition contains the AUTHENTICATE(BASIC) parameter. HTTP Basic authentication is used to obtain a user ID and password from the plug-in. Basic authentication requires the SEC system initialization parameter to be set to YES. If SEC is set to NO, then the TCPIP SERVICE resource with BASIC authentication cannot be installed in the region. If you do not want to use this type of authentication, define AUTHENTICATE(NO) in the CIUDEFW sample file in hlq.SCIUSAMP.CICS before the CIUJCWEB job is run. However, if you use AUTHENTICATE(NO), you can work from the plug-in only as the CICS default user.

For more information about configuring TCP/IP, see *Configuring support for communicating over a TCP/IP network* in the CICS Transaction Server Knowledge Center.

You can now recycle the CICS region.

6. Check the web service installation results.

When the controlling region is started, after the successful completion of the CIUJCWEB job, you can check the resource definitions by using the **CEMT INQUIRE PIPELINE** and **CEMT INQUIRE TCPIP SERVICE** commands. If the resource installation was successful, you can now use the web service. If you experience any problems, check the results of the CIUJCWEB job and the resource definition details. Also, check that the WEBSERVICE and URIMAP resources were created. These resources are created automatically during the PIPELINE resource installation. You can issue the **CEMT PERFORM PIPE(CIUWPIPE) SCAN** command that re-creates the WEBSERVICE and URIMAP resources from the pipeline definition. The expected response code for this command is NORMAL.

Run the IVP transaction to verify the installation

Use the CICS IA installation verification program (IVP) to verify that CICS IA is installed correctly in your CICS region. You are recommended to fix any errors reported by the IVP before you run any other CICS IA program.

About the IVP

The installation verification program checks CICS RDO definitions to ensure that all the 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. The messages show verified elements and also any elements of CICS IA that are missing or unavailable. The lack or unavailability of an element can 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 319. Error messages give the probable cause of the error and the action to take to correct it.

Running the IVP

To run the IVP, follow these steps:

1. Log on to the CICS region where installed CICS IA is installed.
2. Clear the CICS screen.
3. Type the transaction name CIUT.
4. Press Enter to start the program. The program might take several minutes to validate the software.
5. When the verification program is complete, check the messages that are 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 see, Chapter 4, “Running the Collector,” on page 85.

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

- Check all the messages 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 that are issued by the IVP are in the range CIU1001 through CIU1013 and are listed in “Messages that CICS IA can issue” on page 319.

- Locate any missing resources that are 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).
- Run the IVP again until it confirms that CICS IA is installed correctly.

Upgrading an existing DB2 configuration

Before you begin upgrading your DB2 configuration, ensure that you have read and followed the instructions to prepare your CICS IA environment and the DB2 considerations.

For more information, see “Prepare your CICS IA configuration environment” on page 31 and “DB2 considerations” on page 35.

There are two methods to upgrade the DB2 tables:

Copy the previous CICS IA DB2 tables to a new database

If you choose to use this method, you must first create a new CICS IA DB2 database and then use DB2 utilities to UNLOAD from the previous database and LOAD into the new database.

Upgrade the previous CICS IA DB2 database to the new structure

If you choose this method, you update only the tables that have a different structure from the tables in your previous CICS IA version. This method is a single database migration method.

Note: CICS IA V5.3 does not provide jobs to port your existing collector VSAM files. You must load all of your VSAM data in to your existing DB2 tables before you proceed with the DB2 upgrade. CICS IA V5.3 provides the upgrade job for the Command Flow data from CICS IA V5.2 only.

In this section:

- Configure an existing DB2 environment for upgrade.
- Upgrade an existing DB2 environment.

Related concepts:

“Prepare your CICS IA configuration environment” on page 31

CICS IA uses an ISPF dialog that runs REXX execs to assist you in configuring your CICS IA environments.

“DB2 considerations” on page 35

Before you run the installation jobs to create the DB2 environment, you must

consider and decide the following questions.

Configuring an existing DB2 environment

To select an existing DB2 environment select option 2 from the configuration menu.

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

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

Please select an option from the list below:

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

It is recommended that you take a copy of the configuration you want to upgrade. Enter c against the region you want to upgrade. This action creates a new configuration, the first two letters of the configuration name is replaced with c_.

```
***** CICS Interdependency Analyzer for z/OS - V5R3M0 Row 1 to 3 of 3 *****
Command ==>                               Scroll ==> CSR

                                CICS IA DB2 Customization Function

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

For selecting a DB2 configuration from the list below, please type
S for Short configuration, F for Full configuration, D for Delete,
C for Copy or R for Rename.

Cmd  Configuration  DB2Version  Description
   s  c_TIA53        V910          CICSIA 53 Database
      DB2V11         V11           DB2 V11

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

Select the copied region and follow the on-screen instructions.

You can select option S (Short) to configure the minimum number of variables or option F (Full) to configure the full set of variables.

DB2 configuration

For a full description of all the configuration values refer to the list in, “Configure a new DB2 environment” on page 38.

To upgrade, you must review and update the following fields:

CONFIGURATION NAME

The name of the new or upgraded DB2 configuration. Set this field to a meaningful name, like the name of the DB2 subsystem in which you are creating the CICS IA DB2 database.

REQUIRED.

DESCRIPTION

A short description of the DB2 configuration.

OPTIONAL.

Output data sets

It is recommended that you change these values to reflect the new version of CICS IA you are upgrading to, and to preserve your previous version copies.

DB2 Variables

If you chose to use the second upgrade method, “Upgrade the previous CICS IA DB2 database to the new structure” on page 63, do not change these values.

If you chose to use the first upgrade method, “Copy the previous CICS IA DB2 tables to a new database” on page 63 you must change the following fields to reflect the new version of CICS IA to which you are upgrading.

- DB2 DATABASE FOR TBLSPCS
- DB2 TABLE QUALIFIER
- DB2 PLAN NAME FOR BATCH
- DB2 COLLECTION ID

For a full description of all these values, see the list in the DB2 Configuration in “Configure a new DB2 environment” on page 38.

Migration variables for DB2

IA PRODUCT QUALIFIER

The CICS IA data set high-level qualifier. This variable is passed in as input when you start the exec.

PREVIOUS RELEASE

Your previous CICS IA release number. CICS IA configures the sample jobs that are required to upgrade your DB2 data to the CICS IA V5.2 database. You can upgrade only from CICS IA 5.1 or CICS IA 3.2.

OPTIONAL.

OLD DB2 TABLE QUALIFIER.

Your previous value for DB2 TABLE QUALIFIER. This variable is used if you keep your previous database and upgrade the data with the samples that are provided to your new V5.2 database.

This field is REQUIRED if you are defining a previous release.

Creating the upgraded DB2 environment

This section describes the steps that are required to define your CICS IA DB2 database and how to review and submit the sample jobs that you configured in the previous section.

When the configuration utility runs to update your DB2 environment variables, you should have the following output data sets.

- hlq.SCIUSAMP.DB2
- hlq.SCIUSQL.DB2
- hlq.SCIUDAT1.DB2
- hlq.SCIUDAT2.DB2

The data set hlq.SCIUSAMP.DB2 contains the sample jobs that are required to define, upgrade, and maintain your CICS IA DB2 environment.

If you chose to use the first upgrade method “Copy the previous CICS IA DB2 tables to a new database” on page 63, perform the tasks that are described in “Step - 1A Creating and updating the IA DB2 database,” and then proceed to “Step 1C - Updating the stored procedures” on page 67.

If you chose to use the second upgrade method, “Upgrade the previous CICS IA DB2 database to the new structure” on page 63, perform the tasks that are described in “Step 1B - Updating the IA DB2 database” and then proceed to “Step 1C - Updating the stored procedures” on page 67.

Step - 1A Creating and updating the IA DB2 database

To create the CICS IA database, perform the following steps:

- Review sample job CIUDBCQ in hlq.SCIUSAMP.DB2.
- Review the associated SQL member, CIUMAIN, in hlq.SCIUSQL.DB2.
- Run sample job CIUDBCQ to create the database objects. This job completes with a return code of 0.

To copy the data from your previous CICS IA database, perform the following steps:

If you are upgrading from CICS IA V3.2:

- Review and run sample job CIUMIG32 to copy the data. This job completes with a return code of 0.

If you are upgrading from CICS IA V5.1:

- Review and run sample job CIUMIG51 to copy the data. This job completes with a return code of 0.

If you are upgrading from CICS IA V5.2:

- Before upgrading the Command Flow data, ensure that APAR PI29675 is installed in your CICS IA V5.2 system. If this APAR is not installed, install APAR PI29675.
- Review and run sample job CIUMIG52 to copy the data. This job completes with a return code of 0.

Step 1B - Updating the IA DB2 database

To update your existing CICS IA database, perform the following steps:

If you are upgrading from CICS IA V3.2:

- Review and run sample job CIUNMG32 to update the database structure. This job completes with a return code of 0.

If you are upgrading from CICS IA V5.1:

- Review and run sample job CIUNMG51 to update the database structure. This job completes with a return code of 0.

If you are upgrading from CICS IA V5.2:

- Before upgrading the Command Flow data, ensure that APAR PI29675 is installed in your CICS IA V5.2 system. If this APAR is not installed, install APAR PI29675.
- Review and run sample job CIUNMG52 to copy the data. This job completes with a return code of 0.

If the upgrade is successful, you can delete the temporary files:

- Review and run sample job CIUNMGQD to update the database structure. This job completes with a return code of 0

Note: The default table space size for tables that are created in the DSN1TIAD step of the CIUNMG32, CIUNMG51, or CIUNMG52 jobs might not suit your system requirements. If they do not, change the size in the corresponding SQL files that are contained in the hlq.SCIUSQL library.

Step 1C - Updating the stored procedures

If you are updating the database on DB2 V9 or later, follow these steps:

- Review the sample job CIUDBCP1 in hlq.SCIUSAMP.DB2.
- Review the associated SQL member, CIUSPDL1, in hlq.SCIUSQL.DB2 and remove the comments to DROP the Stored Procedures.
- Run sample job CIUDBCP1 to create the database objects. This job completes with return code 0.

If you are updating the database on DB2 V8 or earlier, follow these steps:

- Review the sample job CIUDBCP2 in hlq.SCIUSAMP.DB2.
- Review the associated SQL member, CIUSPDL2, in hlq.SCIUSQL.DB2 and remove the comments to DROP the Stored Procedures.
- Run sample job CIUDBCP2 to create the database objects. This job completes with return code 0.

Step 2 - Rebind the IA DB2 packages

To bind the DB2 batch packages that are required by the CICS IA database, follow these steps:

- Review the sample job CIUDBNB in hlq.SCIUSAMP.DB2.
- Review the associated SQL member, CIUGRNTB, in hlq.SCIUSQL.DB2. This sample member includes sample SQL to grant a RACF group or user ID access to the CICS IA plans. It is used in the STEP 3 of the sample job.
- Run the sample job CIUDBNB to bind the DB2 batch programs that are used by CICS IA sample jobs and to grant access to the bound plan. This job completes with a return code 0.

Step 3 - Loading static DB2 tables

CICS IA uses the following static tables:

CIU_VERSION

The CIU_VERSION table contains the CICS IA Version and Service information and is used by the CICS IA plug-in for CICS Explorer.

- To load the version table, review and run job hlq.SCIUSAMP.DB2(CIUVERLD). This job completes with a return code 0.

CIU_TRANSLATORS

The CIU_TRANSLATORS table contains a one to one relationship between the IBM program component numbers for compilers, translators, and linkage editors and a description for each component.

- To load the translator table, review and run job hlq.SCIUSAMP.DB2(CIUTLOAD). This job completes with a return code 0.

CIU_THREADSafe_CMD

The CIU_THREADSafe_CMD table contains information on whether an EXEC CICS command is threadsafe or not for each version of CICS TS. It is used by the threadsafe report program and the stored procedure to determine whether a command is threadsafe.

- To load the threadsafe table, review and run job hlq.SCIUSAMP.DB2(CIUTSLOD). This job completes with a return code 0.

CIU_TRUEEXIT_INFO

The CIU_TRUEEXIT_INFO table contains a one to one relationship between the CICS TRUEs that are used in your environment and a description for each TRUE. This table is used by the CICS IA plug-in for CICS Explorer.

- To add your own TRUEs to the table, edit and save hlq.SCIUDAT2.DB2(CIUTRCD).
- To load the TRUE exit table, review and run job hlq.SCIUSAMP.DB2(CIUTRLOD). This job completes with a return code 0.

Step 4 - Loading the CIU_CICS_CHAINP table (for DB2 V9 onwards)

To load the CIU_CICS_CHAINP table, complete the following steps:

- Review the sample job CIUCHPFL in hlq.SCIUSAMP.DB2.
- Run the sample job CIUCHPFL. This job completes with a return code 0.

Step 5 - DB2 Stored Procedures Setup

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

If you are running DB2 V910 or later, CICS IA uses Native DB2 stored procedures and these stored procedures are defined in STEP 1C.

If you are running DB2 V810 or earlier, CICS IA uses external stored procedures and these stored procedures are defined in STEP 1C.

External DB2 stored procedures run in a started task that is called WLM (Workload Manager) associated with each DB2 subsystem. Sample JCL for a WLM started task is in hlq.SCIUSAMP.DB2(CIUSPTSK). If you already have a DB2 WLM task, you need to update to change the CICS IA load library hlq.SCIULOAD in the STEPLIB concatenation in this JCL to the latest release.

Note: The name of the started task must match the one that is supplied in the CICS IA configuration variable DB2 WLM PROCEDURE NAME.

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

Upgrading your collection configuration

How to configure an existing collection environment for upgrade, upgrade an existing collection environment, and run the IVP transaction to verify the installation.

Before you begin upgrading your collection configuration, ensure that you read and follow the instructions in, “Prepare your CICS IA configuration environment” on page 31. Also, review “CICS considerations” on page 46.

Configuring an existing collection environment

To select an existing collection configuration, select option 4 from the configuration menu.

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

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

Please select an option from the list below:

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

Take a copy of the configuration you want to upgrade. Enter the letter c against the region you want to upgrade. This creates a new configuration and the first two letters of the configuration name are replaced with c_.

```
***** CICS Interdependency Analyzer for z/OS - V5R3M0 Row 1 to 6 of 6
Command ==>                               Scroll ==> CSR

                                CICS IA CICS Customization Function

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

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

Cmd  Configuration  CICSVersion  Description
s    c_3INC09       V530        V53INC09
      MYTEST        V520
      V53INC09      V530        V53INC09

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

Select the copied region and follow the on-screen instructions.

You can select option S (Short) to configure the minimum number of variables or option F (Full) to configure the full set of variables.

CICS configuration

For a complete description of all the configuration values, see list in the CICS Configuration in “Configuring a new collection environment” on page 50.

To upgrade, review and update the following fields:

CONFIGURATION NAME

The name of the new or upgraded CICS configuration. Set this field to a meaningful name, like the name of the CICS job, the CICS APPLID, or a CPSM CICS group definition.

REQUIRED.

DESCRIPTION

A short description of the CICS configuration.

OPTIONAL.

Output data sets

Change these values to reflect the new version of CICS IA you are upgrading to, and to preserve your previous version copies.

CICS IA variables

Change the file qualifiers to reflect the new version of CICS IA you are upgrading to and to preserve your previous version copies.

CICS IA command flow variables

Change the file qualifiers to reflect the new version of CICS IA you are upgrading to and to preserve your previous version copies.

For a full description of these values, see the list in “Configuring a new collection environment” on page 50.

CICS variables

Create or update the CSD group names to reflect the new version of CICS IA to which you are upgrading.

Explorer plug-in variables.

For a description of the Explorer plug-in values, see “CICS Explorer plug-in variables” on page 56.

Migration variables for the control file

IA PRODUCT QUALIFIER

The CICS IA data set high-level qualifier. This variable is passed in as input when you run the exec.

PREVIOUS RELEASE

Your previous CICS IA release number. CICS IA then configures the sample jobs that are required to upgrade your CICS IA Control File. You can upgrade only from CICS IA V3.2 or CICS IA V5.1.

Note: CICS IA V5.3 does not support upgrading an existing CICS IA Control file.

OPTIONAL.

PREVIOUS IA VSAM QUALIFIER

The CICS IA data set qualifier that is used for the allocation of the VSAM files that are used in the previous version.

This field is **REQUIRED** if you are defining a previous release.

Creating the upgraded collection environment

How to define the resources that are required to enable CICS IA collection in your CICS regions, review and submit the sample jobs that you configured previously.

When you run the configuration utility to define your CICS environment variables, the following output data sets are created.

- hlq.SCIUSAMP.CICS
- hlq.SCIUSAME.CICS
- hlq.SCIUSQL.CICS
- hlq.SCIUDAT1.CICS

Note: If you configured the CICS IA collector to be used with a UDB database, then SCIUSQL and SCIUDAT1 are not created

The data set hlq.SCIUSAMP.CICS contains the sample jobs that are required to define and maintain your CICS IA collector environment.

Step 1 - Defining the VSAM files

To create the CICS IA VSAM files that are used by CICS IA, perform the following steps:

To upgrade the CICS IA Control file:

- To upgrade from CICS IA V3.2, review and submit the sample job in hlq.SCIUSAMP.CICS(CIUUP32C).
- To upgrade from CICS IA V5.1, review and submit the sample job in hlq.SCIUSAMP.CICS(CIUUP51C).
- To upgrade from CICS IA V5.3, review and submit the sample job in hlq.SCIUSAMP.CICS(CIUUP52C).

These jobs complete with a return code of 0.

To define the CICS IA collector files:

- Review and submit sample job CIUJCLCA in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 0.

Note: You can not upgrade your collector files from previous releases. Jobs are provided to upgrade your DB2 environment to CICS IA V5.2 to ensure that all your collected data is loaded into DB2 tables.

Step 2 - Defining the Command Flow resources

The CICS IA command flow feature uses CICS logstreams and GDGs to facilitate the offloading of captured data into the DB2 tables. If you have decided to use new qualifiers for this version of CICS IA, you need to perform the following steps.

To create these resources, perform the following steps:

To define the logstream that is used by CICS IA:

- Review and submit sample job CIUJCDLS in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 0.

To define the user GDG data sets that are used by CICS IA:

- Review and submit sample job CIUJCLCG in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 0.

Note: You must define GDG data sets for every CICS user ID that needs to use the Command Flow feature

Step 3 - Updating the IA resources to CICS

The CICS resources that are required by CICS IA change from release to release. It is recommended that you delete the CICS resources and the CSD groups before you define the new ones.

To remove old CICS resource definitions:

- Review and submit the sample job CIUDELGR found in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 0.

Note: If you selected new group names for this upgrade, you must update the group names in this sample.

To define the main CICS resources that are required by CICS IA:

- Review and submit the sample job CIUJCINT in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 4, which is acceptable.

This job is configured to include all the resources that are required to be defined based on your selections when you run the configuration exec. For example, if you selected to use LIBRARY names then these definitions are included. This job also defines the resources that are required for NLS support. The CSD and group names that are selected in the configuration exec are used.

To define the VSAM file resources that are required by CICS IA:

If you are using a single CICS region, or you are using VSAM RLS to share the files, review, and submit the sample job CIUJCFIL in hlq.SCIUSAMP.CICS.

If you are using an FOR to share the files, then review and submit the following jobs:

- hlq.SCIUSAMP.CICS(CIUJCFIL) to define the CICS file resources locally or in a file-owning region.
- hlq.SCIUSAMP.CICS(CIUJCFIR) to define the CICS file resources as remote to a file-owning region.
- These jobs complete with a return code of 0.

To define the web service resources that are required by CICS IA:

Review and submit the job CIUJCWEB in hlq.SCIUSAMP.CICS. This job completes with a return code of 0.

If you previously used Atomservice to work with the CICS IA plug-in, remove the Atomservice CSD group from the CICS IA controlling region before you install the web service CSD group. You can use the

hlq.SCIUSAMP.CICS(CIUDELGR) job to remove the Atomservice CSD group. If the web service is going to use the same TCP/IP port, which was previously used by Atomservice, then Atom resources must be deleted.

Step 4 - Updating the DB2 resources

If you previously collected DB2 resource information in your target CICS region follow these steps.

To bind and package the DB2 plans that are required for DB2 collection:

- Review and submit the sample job CIUDBNT in hlq.SCIUSAMP.CICS.
- This job completes with a return code of 0.

Note:

- If you selected to configure CICS IA to collect DB2 resources, then the DB2 resources for DB2TRAN and DB2ENTRY are defined in STEP 3.
- On any region where you want to collect DB2 data, ensure that the user ID, under which CICS IA runs, has GRANT permission to the batch plan that is created in the sample job SCIUSAMP.CICS(CIUDBNT). The GRANT permission enables the background transaction CINB to access the SYSIBM.SYSDUMMY1, SYSIBM.SYSPACKSTMT, and SYSIBM.SYSSTMT DB2 tables.

Step 5 - Tailoring your CICS startup job.

To enable CICS IA to run in your CICS region, tailor your CICS startup JCL as follows:

If you did not select to use the LIBRARY definitions or the CICS region is CICS TS V3.2 or earlier, then you must update the following load libraries in the DFHRPL concatenation in the startup job JCL to point at the latest CICS IA libraries:

- hlq.SCIULOAD
- hlq.SCIULODE, the default, English, national language: this library is always required.
- Add hlq.SCIULODK, if you require Japanese national language support.

Run the IVP transaction to verify the installation

To verify your installation it is recommended that you run the IVP program as described in “Run the IVP transaction to verify the installation”.

“Run the IVP transaction to verify the installation” on page 62.

Post configuration tasks

This section describes some optional tasks that you might be required to be perform.

Creating your own program exclude, transaction exclude, command exclude, and resource compression list

You can use program exclude lists, transaction exclude lists, command exclude lists, and resource compression 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 but for command exclude list the term, Collector, refers only to the Dependency collector functions.

For details of how to specify command flow exclude information, see “Managing the Command Flow collection using transaction CINC” on page 125.

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. Similarly, a command exclude list contains a list of CICS command codes and groups of command codes. The Collector does not collect data for any command or any command from the command group.

For details of how to specify which exclude lists the Collector is to use, see “Specifying region-specific options: General” on page 108.

CICS IA supplies sample program exclude, transaction exclude, command exclude, and resource compression 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 command exclude list, supplied with CICS IA, is called CIUXCOMM and is empty.
- The default resource compression list, supplied with CICS IA, is called CIUXRCOM 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. You can use the provided sample batch job to create and edit the sample program exclude list.

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 6 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 6. Example program exclude list

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 h1q.SCIUSRCE, where “h1q” 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 h1q.SCIULOAD, where “h1q” 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 7 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 7. 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 hlq.SCIUSRCE, where "hlq" 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 hlq.SCIULOAD, where "hlq" 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 command exclude list

A command exclude list is a load module that contains a simple list of command codes and group command codes.

About this task

Each list item consists of a 1-byte length field that contains the length of the command code or the group command code. A CICS command code or group command code follows the length field. Both codes are in hexadecimal form. The length field is always equal to 2. The Group command code has a zero in the second byte. A length of zero indicates the end of the list.

Figure 8 is an example of a command exclude list.

MYXCOMM	CSECT	
MYXCOMM	AMODE	31
MYXCOMM	RMODE	ANY
	DS	0F
DC	AL1(2),X'0802'	Excludes WRITEQ TDQUEUE command
DC	AL1(2),X'0804'	Excludes READQ TDQUEUE command
DC	AL1(2),X'0C00'	Excludes all commands from group 0C
DC	AL1(0)	End of list
END	MYXCOMM	

Figure 8. Example command exclude list

A sample batch job, CIUJCLXC, is provided to assemble and link-edit the sample command exclude list, CIUXCOMM.

Before running the CIUJCLXS 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 command 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 command exclude list.

2.

To make your customized command 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, by using the same attributes as those used for CIUXCOMM in the CIUJnnCR sample JCL in the CICS IA load library. In particular, specify RELOAD(NO) on the PROGRAM definition).

Creating a resource compression list

A resource compression list is a load module that contains a simple list of CICS TSQueue, CICS CHANNEL/CONTAINER, CICS ENQ/DEQ and Websphere MQ resource name keys. You can use the provided sample batch job to create the sample or your own resource compression list.

The purpose of the Resource compression list is to compress records in the CICS IA Dataspace to reduce memory usage or avoid the output of unnecessary data into the DB2 tables. Records that match the rules that are specified in this resource list are collected as just one record. If a match is found then the rest of the resource name is changed to contain “+” plus signs.

There are three versions of resource compression list format. The higher the version is, the more precision in the compression of records is available. Resource compression rules of different formats can not exist in a same list, they are incompatible with each other. For all versions, fields can not expand out of the 32 byte range. The end of the list is always indicated by a binary zero byte.

Compression rule format version 1

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

```

    TSQ name = DEPT10VT27SPA
    altered to = DEPT10++++++

    ENQ name = FXTRAN
    altered to = FX++++

    MQ  name = QUE0001
    altered to = QUE++++

    CIUXRCOM CSECT
    CIUXRCOM AMODE 31
    CIUXRCOM RMODE ANY
    DC    AL1(2),C'FX'
    DC    AL1(3),C'QUE'
    DC    AL1(6),C'DEPT10'
    DC    AL1(0)                                End of list
    END    CIUXRCOM

```

Figure 9. Example of the compression rule format version 1

Compression rule format version 2

This format of the resource compression list rule allows a string to be specified at any fixed starting position within a resource name of a selected resource type. The string can therefore be a prefix, an infix, or a suffix. Such a string is called the key.

The rule format is the following:

- CICS TS Command type :
 - decimal 1 = MQ
 - decimal 4 = CHANNELs/CONTAINERs
 - decimal 7 = ENQ/DEQ
 - decimal 10 = TS Queue
 - decimal 50 = TS Queue, (affinity)
- Starting position of the key, range 1 - 32
- The key length, range 1 - 32
- The key, which is a string of valid symbols

```

TSQ name = T520ACCTBAL
altered to = +++ACCT+++

TSQ name = SR-AREA01
altered to = SR+++++++

ENQ name = EURPAY
altered to = +++PAY

MQ Queue name = FRQUE001
altered to = ++QUE++

CIUXRCOM CSECT
CIUXRCOM AMODE 31
CIUXRCOM RMODE ANY
DC    AL1(7),AL1(4),AL1(4),C'PAY'    ENQ/DEQ Example
DC    AL1(10),AL1(5),AL1(5),C'ACCT'  TSQ Example
DC    AL1(10),AL1(1),AL1(2),C'SR'    TSQ Example
DC    AL1(1),AL1(3),AL1(3),C'QUE'    MQ Example
DC    AL1(0)                          End of list
END    CIUXRCOM

```

Figure 10. Example of the compression rule format version 2

Compression rule format version 3

This format is an extension of the resource compression list rule format version 2, but without Affinity support. Two additional options are added:

Preservation area

Defines part of a resource name that is not filled with “+” pluses. This part can include the key.

Compression trigger

The amount of records at which the compression rule is applied and compression starts. If the amount of records is greater than one, such a rule is only applicable to looped CICS TS API calls, because this amount is associated with the combination of application, transaction, program, offset, and other key fields, depending on a resource type.

Note: If the compression trigger is greater than 1, then the union of the key and preservation area must be continuous and start from the first position in a resource name, otherwise there is a list format error. This restriction exists because of performance considerations.

Before adding rules, you must first specify the eyecatcher so that CICS IA knows that version 3 of the resource compression list is used: DC CL8'PLIST V3'

The rule format is the following:

- CICS TS Command type
 - decimal 1 = MQ
 - decimal 4 = CHANNELS/CONTAINERS
 - decimal 7 = ENQ/DEQ
 - decimal 10 = TS Queue
- Compression trigger, range 1 - 255
- Starting position of the preservation area, range 1 - 32
- The preservation area length, range 0 - 32
- Starting position of the key, range 1 - 32

- The key length, range 0 - 32
- The key which is a string of valid symbols

Initial conditions:

```
TSQ name = TSQ*****
where ***** increments by 1 from 00001 to 25000
at each CICS TS API call
```

Compression list:

```
CIUXRCOM CSECT
CIUXRCOM AMODE 31
CIUXRCOM RMODE ANY
DC    CL8'PLIST V3'
DC    AL1(10),AL1(3),AL1(4),AL1(1),AL1(1),AL1(3),C'TSQ'
DC    AL1(0)                                     End of list
END    CIUXRCOM
```

CICS IA output:

```
TSQ name = TSQ00001
TSQ name = TSQ00002
TSQ name = TSQ00+++
TSQ name = TSQ10000
TSQ name = TSQ10001
TSQ name = TSQ1++++
TSQ name = TSQ20000
TSQ name = TSQ20001
TSQ name = TSQ2++++
```

Note 1:

```
DC AL1(10),AL1(3),AL1(4),AL1(1),AL1(1),AL1(3),C'TSQ'
is equivalent to
DC AL1(10),AL1(3),AL1(1),AL1(4),AL1(1),AL1(3),C'TSQ'
```

Note 2:

```
DC AL1(10),AL1(1),AL1(3),AL1(3),AL1(3),AL1(3),C'TSQ'
is equivalent to
DC AL1(10),AL1(3),AL1(3),C'TSQ' in rule format ver. 2
```

Figure 11. Example of the compression rule format version 3

Grouping transactions and programs into applications

CICS IA Version 5.1 and later provides utilities so that you can use CICS IA Version 3.2 Application definitions in your CICS IA Version 5.1 or later environment. However, CICS Transaction Server Version 5.1 and later provides the capability to define applications as part of the platform capabilities, and it is now advisable to define applications in this way.

In CICS IA Version 3.2, you could group transactions, programs, or both to form a notional *Application*. CICS IA Version 5.1 and later provides utilities so that you can continue to use these Application definitions.

However, it is now advisable to use the function introduced in CICS TS Version 5.1 to define applications as part of the platform capabilities. You can logically define the resources that comprise a business application in CICS TS as a single entity, and deploy these resources to CICS TS as a single resource. An application that is defined in this way can be managed as a single entity throughout its lifecycle, making CICS TS application management faster, simpler, and less error prone. For more information, see Platforms in the CICS TS Knowledge Center.

Migrating application definitions

You can use your CICS IA Version 3.2 Application definitions in your CICS IA Version 5.1 or later environment by reloading the XML files you created in CICS IA Version 3.2: v32hlq.SCIUDAT2.OUT(CIUAPPLS). To do this, review and run the sample job in hlq.SCIUSAMP.DB2(CIUALOAD). You must use the CICS IA Version 3.2 data set as input to CIUMIGXT DD statement in the second step.

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 105.

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 RCA , RDCEXIT and RDCSIZE parameters in the Natural/CICS parameter module. These parameters are options on the NTPRM macro. Set the parameters as follows:

- RCA=(CIURDCX1)
- RDCSIZE=2
- RDCEXIT=(CIURDCX1,400)

Later versions of Natural do not support the use of RDCEXIT as a parameter on the NTPRM macro. Instead you must use the NTRDC macro as shown:

- NTRDC EXIT=(CIURDCX1,400)

You can also specify the RDCSIZE using the NTDS macro as shown:

- NTDS RDCSIZE,2 DATA Collector Buffer

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=h1q.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=h1q.SCIULOAD
```

```
INCLUDE SCIULOAD(CIURDCX1)
```

What to do next

When these steps are complete 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 one task: Modifying the Exclusive Work Area (EWA) 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.

Starting and stopping CICS IA from the PLT

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

About this task

You can stop CICS IA from a program list table (PLT) program that is initiated during CICS termination; that is, a program that is specified in the first part of the program list table shutdown (PLTSD) list for the CICS region.

Note: If you choose to start CICS IA during PLT startup, it is NOT a requirement to stop CICS IA during PLT shutdown.

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

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

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 lines 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*.

CICS IA supplied modules required in the MVS link list

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 link list.

Creating your translation table

CICS IA uses a translation table to translate the resource name of the CICS IA dependency data before it is loaded to the DB2 CIU_CICS_DATA table or it creates the *hlq*.CICS.CSV file.

About this task

By using a translation table, each "XY" byte of the resource name is converted to a byte whose offset from the start of this table is "XY". The translation table is a load module that has a fixed name of CIUTRNTB and a fixed location in the *hlq*.SCIULOAD library, where *hlq* is the high-level data set qualifier that was assigned during the CICS IA® installation.

You can control the translation by using the parameter that is passed to the CIUU040 module in the following sample jobs:

- CIUUPDB
- CIUUPDB1
- CIUUDB

Specify Y to translate the data or N not to translate the data.

CICS IA supplies a sample program translation table. The default program translation table, that is supplied with CICS IA, is a load module called CIUTRNTB. The default translation table converts non-printable characters to ".".

A sample batch job, CIUJCLTT, is provided to assemble and link-edit the sample program translation table, CIUTRNTB. You can use the provided sample batch job to create your own translation table.

Procedure

1. Copy the sample CIUJCLTT job before you edit the JCL.
2. Change the JOB account and JOB card parameters to reflect your local environment.
3. Update the PGM keyword of the EXEC statement in the ASM step and enter the name of the assembly language that you want to use.
4. Modify the SYSIN DD statement and specify the name of the assembly language source library where your translation table can be found. The default library is *hlq.SCIUSRCE*, where *hlq* is the high-level data set qualifier that is assigned during the CICS IA installation. Change the member name to the name of your translation table.
5. You do not need to change the SYSLMOD DD statement. The translation table is a load module that must be called CIUTRNTB in the *hlq.SCIULOAD* library, where *hlq* is the high-level data set qualifier that is assigned during the CICS IA installation.
6. When you are finished, you can submit the job to create your translation table. Ensure that the job completed successfully.

Chapter 4. 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 86
- “Controlling the collection of dependency and affinity data” on page 94, which contains:
 - “Starting data collection” on page 95
 - “Changing the data collection options dynamically” on page 97
 - “Pausing the collection of data” on page 99
 - “Resuming the collection of data” on page 100
 - “Stopping the collection of data” on page 101
 - “Displaying Collector statistics for a specified region” on page 102
- “Changing the Collector options” on page 105
- “Collector errors” on page 209

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 94.
- Change its run time options, which is described in “Changing the Collector options” on page 105.

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 Collector Main Administration Menu panel, CIU000, to:

- Specify at least one CICS region to be monitored
- Specify (in further screens) the CICS, CICSplex SM, 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 105.

If you have migrated the CIUCNTL control file from a previous CICS IA release, some CICS regions might already be configured for monitoring.

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 Collector Global Options Menu panel, CIU300.

Displaying the Collector Main Administration Menu panel

Use this procedure to display the Collector Main Administration Menu panel.

About this task

To display the Collector Main Administration Menu panel:

1. At a CICS terminal, type the transaction identifier CINT and press Enter.
2. Panel CIU000 is displayed. See Figure 12. 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 - V5R1M0          2012/10/01
                  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 5655-Y22 (C) Copyright IBM Corp. 2001, 2014  6
F1=Help    F2=          F3=Exit    F4=          F5=          6
F7=        F8=          F9=        F10=         F11=         7
F12= Exit
```

Figure 12. 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 5655-Y22 (C) Copyright IBM Corp. 2001, 2014
- 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.

Command Flow User Administration using transaction CINT

You can use the Command Flow User Administration using transaction CINT feature to view and manage the accounts of the Command Flow collector users. This function 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, or 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 Collector Main Administration Menu panel, CIU000.
3. On panel CIU000, choose option 4.
4. Press Enter. The User Administration Menu panel, CIU400, is displayed. See Figure 13 on page 88.

CIU400

CICS Interdependency Analyser for Z/OS - V5R3M0

2014/12/02

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

Act	CINC User ID	CINC USER STATUS	CINC AUTHORITY	Act	CINC User ID	CINC USER STATUS	CINC AUTHORITY
—	SSUSERSS	NOT ACTIVE	PRIVILEGED	—			
—	##USER##	NOT ACTIVE	GENERAL	—			
—	@@USER@@	NOT ACTIVE	GENERAL	—			
—	TTTTTTTT	NOT ACTIVE	GENERAL	—			
—	USER1	NOT ACTIVE	PRIVILEGED	—			
—	XXUSERXX	NOT ACTIVE	PRIVILEGED	—			
—	1USER1	NOT ACTIVE	GENERAL	—			
—	33333333	ACTIVE	GENERAL	—			
—				—			

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 13. 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 2.

Table 2. 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 2. 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 14.

CIU410

CICS Interdependency Analyser for Z/OS - V5R3M0

2014/12/02

Add User Menu

12:33:07PM

Specify new user name together with corresponding journal name.
Press ENTER to add new user without leaving this panel.

New user name:

1

CINC Authority:

(Privileged or General)

2

Journal name for trace data . .:

CIUMTJNL

3

CICS Sysid: T41B

CICS Applid: IYDZT41B

TermID: TC47

F1=Help

F2=

F3=Add&Exit

F4=

F5=Refresh

F6= 4

F7=

F8=

F9=

F10=

F11=

F12=Cancel

Figure 14. 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 The CINC user can be defined as a GENERAL user or a PRIVILEGED user. PRIVILEGED users can use wildcards to collect data for more than one user ID. The default value is General.

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

Note: The log stream name in a given journal model must be the same on all regions.

4 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 3.

Table 3. 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 15 on page 91.


```

CIU420      CICS Interdependency Analyser for Z/OS - V5R3M0      2014/12/02
              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
CINC Authority . . . . . : GENERAL      2
Journal name for trace data . . : CIUMTJNL      3

Parameters below will be copied
Traced transactions IDs. . . . . :      4
Traced user USERID . . . . . : XXUSERXX      5
Traced terminal TERMID . . . . . : *      6
Command Flow data ID . . . . . :      7
Dynamic call . . . . . : Y      8
User modifiable exit name. . . . . :      9
Traced regions APPLIDs . . . . . : 10

CICS Sysid: T41B      CICS Applid: IYDZT41B      TermID: TC47
CIU2514I User USER1313 was added.
F1=Help      F2=      F3=Copy&Exit      F4=      F5=Refresh      F6= 11
F7=      F8=      F9=      F10=      F11=      F12=Cancel

```

Figure 15. 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** The CINC user can be defined as a GENERAL user or a PRIVILEGED user. PRIVILEGED users can use wildcards to collect data for more than one user ID. The default value is General.
 - 3** Specify the journal name for trace data. The length of the name is up to 8 characters. The default journal name is CIUMTJNL.
- Note:** The log stream name in a given journal model must be the same on all regions.
- 4** The identifiers of the transactions for which you want to trace the Command Flow data.
 - 5** The user identifier associated with CICS transactions, for which you want to collect the Command Flow data.
 - 6** The terminal identifier associated with CICS transactions, for which you want to collect the Command Flow data.
 - 7** The name of the Command Flow trace that is to be captured.
 - 8** 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.
 - 9** The 8-character user modifiable exit name that you can use to add data to a user Command Flow record.

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

11 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 4.

Table 4. 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 16 on page 93.

CIU440

CICS Interdependency Analyser for Z/OS - V5R3M0

2014/12/14

User Details Menu

02:19:07PM

Options of CINC user DSFSDFFS

CINC Authority : GENERAL

Traced transactions IDs. :

Traced user USERIDs. : DSFSDFFS

Traced terminal TERMIDs. : *

Command flow data ID :

Journal name for trace data. . . : CIUMTJNL

Dynamic call : Y

User modifiable exit name. . . . :

Traced region APPLIDs. :

1

2

3

4

5

6

7

8

9

Last data collection statistics

Collector status : NOT ACTIVE

Collector last start :

Collector last stop. :

10

11

12

CICS Sysid: T41B

CICS Applid: IYDZT41B

TermID: TC60

F1=Help

F2=

F3=Save&Exit

F4=

F5=Refresh

F6=

F7=

F8=

F9=

F10=

F11=

F12=Cancel

13

Figure 16. User Details Menu panel, CIU440

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

- 1** The CINC user can be defined as a GENERAL user or a PRIVILEGED user. PRIVILEGED users can use wildcards to collect data for more than one user ID. The default value is General.
 - 2** The identifiers of the transactions for which you want to trace the Command Flow data.
 - 3** The user identifier associated with CICS transactions, for which you want to collect the Command Flow data.
 - 4** The terminal identifier associated with CICS transactions, for which you want to collect the Command Flow data.
 - 5** The name of the Command Flow trace that is to be captured.
 - 6** Specify the journal name for trace data. The length of the name is up to 8 characters. The default journal name is CIUMTJNL.
- Note:** The log stream name in a given journal model must be the same on all regions.
- 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 records.
 - 9** The 8-character APPLID of the CICS region for which you want to trace the Command Flow data.

10 The Command Flow collector status for the specified user.

11 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.

12 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.

13 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 5.

Table 5. 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 Collector Main Administration Menu panel, CIU000). The Collector Operations Menu screen, CIU100, shown in Figure 17 on page 95, 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 - V5R1M0      2012/09/26
                  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  2012/09/25  04:56:07PM  Dependencies
-   IYCLZC04 TLS4  STOPPED
-   IYCLZC05 TLS5  RUNNING  2012/09/26  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 17. 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 data collection

If the CICS IA Collector is not already running in a specific region, you can start collecting dependency data, affinity data, or both on that region. Only one CICS IA Collector can run on each CICS region.

To start collecting data on a specified CICS region, use one of the methods shown in Table 6. The Collector records transaction dependencies or affinities until you pause or stop data collection.

Table 6. 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

Note:

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 as follows:

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 panel, 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 `<start_options>` are described in note **3**.
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 `<start_options>` are described in note **3**. 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.

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

1. The start option (such as START, STARTAFF, STARTINT, or STARTALLAFF) that you specify, as listed earlier.
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/CPSM Resource Options screen, CIU250. See Collector CICS(r) Resources Options panel, CIU240 and Collector DB2/MQ/IMS/CPSM Resource Options panel, 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 panel, 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”, “Transaction exclude list”, “Command exclude list”, and “Resource compression list” in “Specifying region-specific options: General” on page 108.

For a complete list of the program commands that are not monitored, see “What the Collector does not monitor” on page 17.

Each time the Collector is started, a new data space is created. You specify its size on the Collector General Options panel, 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, see “Specifying region-specific options: Region configuration” on page 105.

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 a noticeable time until those records are 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 105.
2. Use one of the methods shown in Methods for changing data collection options to start CINT using the updated options.

Table 7. 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

Note:

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

- All CICS options on screens CIU240 and CIU245
- All CICSplex SM, DB2, IMS, or WebSphere MQ options on screen CIU250
- The following General Options on screen CIU260:
 - Data to Collect
 - Inquire for DB2 resources
 - Maintain usage counts
 - Perform periodic saves
 - Program exclude list
 - Restore data on start
 - Resource compression list
 - Transaction exclude list

- Command exclude list
- Transid prefix
- Trigger for CINB start
- 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 8. **1**

Table 8. 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

Note:

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 8 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 120.

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 8 on page 99 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 108.)

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 9. **1**

Table 9. 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 cicsjob, CINT CONTINUE 4
Console	F cicsjob, CINT CONTINUEALL 5
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 9 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 120.

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 9 on page 100 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 10.

Table 10. 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.
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

Note:

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 10 on page 101 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.” 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 Collector Operations Menu screen, `CIU100`), type 5 (the action code for Statistics) against a listed CICS APPLID and press Enter. The Collector Statistics Menu panel, `CIU150`, shown in Figure 18 on page 103, is displayed.

CIU150	CICS Interdependency Analyzer for z/OS - V5R3M0	2014/08/18
	Statistics Menu for	12:40:17PM
CICS Sysid : Z325 CICS Applid : IYDZZ325		
CINT state	: STOPPED	1
Records written last save. : 0	2	4
Total records on file. . . : 17	3	5
	Collecting Dependencies	
	Number of pauses . . . : 0	
	Number of saves. . . . : 0	
Date/time of last start. . .	: 2014/08/18 10:24:21AM	6
Date/time of last save . . .	:	
Date/time of last change . .	:	
Total time RUNNING	(HHHH:MM:SS)	7
Total time PAUSED.	(HHHH:MM:SS)	
Table dataspace name	% full	8
CICS Sysid: Z325 CICS Applid: IYDZZ325 TermID: TC01		
F1=Help	F2=	F3=End
F7=	F8=Page Down	F9=
	F4=	F5=Refresh
	F10=	F11=
		F6=
		F12=
		9

Figure 18. 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.
 - When you stop the Collector, the CINT state changes to STOPPED only after the Collector has saved the dependency or affinity data.
 - 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.
 - When the state is TERMINATED, the Collector was stopped due to an IUZT abend. (An error was detected when CICS IA queries DB2 and option **Terminate on abend** was set to Y.)
- 2** Depending on whether the Collector is configured to collect dependency data, affinity data, or both on this region, the number of records written to the dependency VSAM data files, affinity VSAM data files, or both dependency and affinity VSAM data files, respectively, when data was last saved.
- 3** Depending on whether the Collector is configured to collect dependency data, affinity data, or both on this region, the total number of records in the dependency VSAM data files, affinity VSAM data files, or both dependency and affinity VSAM data files, respectively.
- 4** The number of times that the Collector has been paused since the last time that it was started.
- 5** The number of times that data has been saved since the last time the Collector was started.
- 6** 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 Collector Global Options Menu panel, CIU300.

- 7** The total times that the Collector has been in the running and paused states since it was last started.
- 8** The name and current percentage occupied of the MVS data space being used. This information is not displayed when the Collector is in the STOPPED state.
- 9** 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, some of which might not be appropriate (or selectable) for a given state of the Collector. When dependency data is being collected, you can use the F8 key to display the Collection Statistics Menu panel, CIU151.

Displaying dependency collector statistics for a specified region

You can display information about new resources or dependencies found while running the Dependency Collector for a specified region with the Collection Statistics Menu panel, CIU151.

From the Collector Statistics Menu panel, CIU150 (shown in Collector Statistics Menu panel, CIU150), press the Page Down (F8) key. The Collection Statistics Menu panel, CIU151, shown in Figure 19, is displayed.

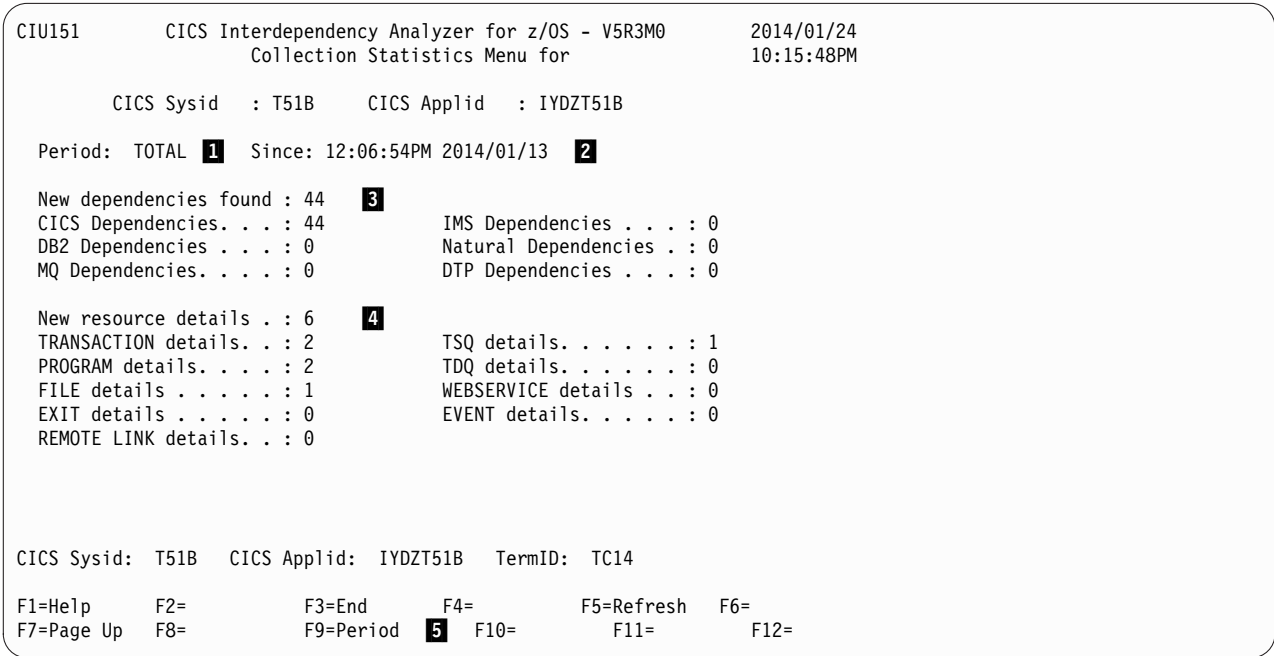


Figure 19. Collection Statistics Menu panel, CIU151

The meaning of each part of the Collection Statistics Menu panel, CIU151, is as follows:

- 1** The length of currently displayed collection statistics period.
- 2** The start time of currently displayed collection statistics period.

- 3** The number of new dependencies found since the last start of the dependency collector.
- 4** The number of new resource details found since the last start of the dependency collector.
- 5** F9 to navigate through periods, this is a looped navigation.

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 123.

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.

The Region Configuration Menu contains a list, which might be empty, of CICS regions to be monitored by CICS IA. If more than one CICS region is in the list, you can use the list item named **ALL** to specify that an operation is applied to all the regions in the list. Also, you can use the list item named **DEFAULTS** to set or change the default values for a specified operation. Initially:

- The values of all the **DEFAULTS** options are set to the CICS IA system defaults.
- The values of all the Collector region-specific options are set to blanks, which means that you use the **DEFAULTS** values.

From the Region Configuration Menu panel, you can:

- Add a CICS region to the list of those to be monitored by CICS IA.
- Delete a CICS region from the list of those to be monitored by CICS IA.
- Copy a CICS region and create a new region with the same resource options as the region being copied.
- Call up a further panel to set or modify some region-specific options for one or for all of the regions being monitored; see “Specifying resource options: Region configuration” on page 107.

CIU200	CICS Interdependency Analyzer for z/OS - V5R1M0 Region Configuration Menu	2012/08/11 17:36:44
--------	--	------------------------

Type action code then press ENTER. More :

1=Add Region 2=Copy Region 3=Delete Region 4=Options **1**

Act	CICS Applid	CICS Sysid	New Applid	New Sysid	Status	Collecting
-	DEFAULTS	DFTS	_____	_____		
-	ALL	ALL	_____	_____		
-	IYDZZ132	Z132			UNCONNECTED	
-	IYDZZ222	Z222			UNCONNECTED	
-	IYDZZ232	Z232			UNCONNECTED	
-	IYDZZ312	Z312			UNCONNECTED	
-	IYDZZ320	MANA			STOPPED	
-	IYDZZ322	Z322			UNCONNECTED	
-						
-						
-						
-						
-						
-						

CICS Sysid: MANA CICS Applid: IYDZZ320 TermID: TC62

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

Figure 20. Collector Region Configuration Menu panel, CIU200

Note:

1. The action codes that you can specify:

1=Add Region

On any line, enter the new region 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.

2=Copy Region

On a line describing an existing region, enter the new region APPLID and 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

To specify the resource options to modify, you use the Resource Options panel. To display this panel, start from the general Region Configuration Menu panel, CIU200, type 4, then press Enter.

The Resource Options panel, CIU290, appears.

CIU290	CICS Interdependency Analyzer for z/OS - V5R3M0	2014/06/22
	Resource Options for	11:01:13AM
	CICS Sysid: ALL CICS Applid: ALL	
Type action code then press ENTER: 1		
1 = General Options		
2 = Time/Date Options		
Interdependency Options		
3 = CICS Options for APIs		
4 = CICS Options for SPIs		
5 = DB2/IMS/MQ/CPSM Options		
6 = Natural Options		
Affinity Options		
7 = Affinity Options		
CICS TS Applications		
8 = Application Data		
CICS Sysid: Z518 CICS Applid: IYDZZ518 TermID: TC45		
F1=Help	F2=	F3=Exit
F4=	F5=	F6=
F7=	F8=	F9=
F10=	F11=	F12=Exit

Figure 21. Collector Resource Options panel, CIU290

You can specify the following action codes:

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” on page 108.

2=Time/Date Options

This option calls up the Time and Date Options panel, CIU280, shown in “Specifying region-specific options: timers” on page 120.

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 113.

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 113.

5=DB2/IMS/MQ/CPSM Options

This option allows you to set or modify which dependency-related CICSplex System Manager, DB2, IMS, and MQ commands are to be monitored on one or

more regions. See “Specifying which dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored” on page 116.

6=Natural Options

This option calls up the Natural Options panel, CIU29N, shown in Collector Natural Options screen, CIU29N.

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 118.

8=Application Data

This option allows you switch Application Data collection on or off and display a selected application for selected region(s).

Specifying region-specific options: General

To specify general region-specific options, you use the Region Configuration Menu panel.

You can use the Region Configuration Menu in the following 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 Collector Region Configuration Menu panel, CIU200). 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 22 on page 109, 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 values of the first “real” CICS region that is listed on the Region Configuration Menu; that is, the values are taken from the first region that is 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, which are 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.

CIU260	CICS Interdependency Analyzer for z/OS - V5R3M0	2015/05/06
	General Options for	02:14:23PM
CICS Sysid : Z53B	CICS Applid : IYDZZ53B	

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

Control options (Fields may be set to blanks for default)

Data to Collect	: I (A=Affinity, I=Interdependency, B=Both)	1
Perform periodic saves	: Y (Y=Yes, N=No)	2
Trigger for CINB start	: 5 (2 to 9999 thousand records)	3
Restore data on start	: Y (Y=Yes, N=No)	4
Maintain usage counts	: Y (Y=Yes, N=No)	5
Size of dataspace	: 16 (10 to 2000 MB)	6
Transid prefix (optional)	: ____ (1 to 4 characters)	7
Program exclude list	: CIUXPROG (1 to 8 characters)	8
Transaction exclude list	: CIUXTRAN (1 to 8 characters)	9
Command exclude list	: CIUXCOMM (1 to 8 characters)	10
Resource compression list	: CIUXRCOM (1 to 8 characters)	11
Dump HLQ	: DUMP (1 to 8 characters)	12
Dynamic call	: (Y=Yes, N=No)	13
Trigger for Task collection	: 7 (1 to 9999)	14
Collect Long Running Tasks	: N (Y=Yes, N=No)	15

CICS Sysid: Z53B CICS Applid: IYDZZ53B TermID: TC68

F1=Help	16	F2=	F3=Exit	F4=	F5=	F6=
F7=	F8=	F9=	F10=	F11=	F12=Cancel	

Figure 22. Collector General Options panel, CIU260

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.

The options that are specified for the Time and Date Options fields override the value that is 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 passed since the last save.
- More than the number of table elements that are specified by the option **Trigger for CINB start** 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 are triggered. The default is 5.

4 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.

5 Maintain usage counts

This field is used to specify whether the Collector records the number of times an interdependency or affinity is detected. The usage count shows 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 is already recorded. When the Collector has been running for some time, and it is detecting fewer and fewer new interdependencies or affinities, fewer and fewer records are 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.

6 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 E, "CICS IA space considerations," on page 387.

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.

7 Transid prefix

The 1- 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 that start with the prefix. If you specify a blank, the default transid prefix in the CICS IA control file is used.

8 Program exclude list

The 1 - 8 character name of a table that contains a list of 1 - 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 that is supplied with CICS IA. For information about how to create your own program exclude lists, see "Creating your own program exclude, transaction exclude, command exclude, and resource compression list" on page 73.

9 Transaction exclude list

The 1 - 8 character name of a table that contains a list of 1 - 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 that is supplied with CICS IA. For information about how to create your own transaction exclude lists, see "Creating your own program exclude, transaction exclude, command exclude, and resource compression list" on page 73.

10 Command exclude list

The 1 - 8 character name of a table that contains a list of 2 character command prefixes. If you specify a blank, the default table name in the CICS IA control file is used. The system default name is CIUXCOMM, which is the name of an command exclude list that is supplied with CICS IA. For information about how to create your own command exclude lists, see "Creating your own program exclude, transaction exclude, command exclude, and resource compression list" on page 73.

11 Resource compression list

The 1 - 8 character name of a table that contains a list of resource compression rules. These rules always include 1 - 32 character resource prefixes. The resource name key is compared to a part of a resource name at the moment it is collected. If a match is found, the other part of the resource name is replaced by "+"s. For example, if your application program uses a TSQueue, which consists of a three character prefix "DFH" and a generated five character numerical value, and if you do not have a rule in the resource compression list for "DFH", CICS IA records all entries for the EXEC CICS commands with all the possible TSQueue resource names: DFH00001, DFH00002, DFH00003... DFHnnnnn. This redundancy might lead to many thousands of entries for the same EXEC CICS command. When the "DFH" key is added to the list, CICS IA can report only one entry for the EXEC CICS command with a resource name of DFH+++++. The system default name CIUXRCOM is the name of a resource compression list that is supplied with CICS IA. For information about how to create your own resource compression lists, see "Creating your own program exclude, transaction exclude, command exclude, and resource compression list" on page 73. See also "ENQ/DEQ" on page 241 and "TS commands" on page 241 to learn about these resources.

12 Dump HLQ

This field is used to define the high-level qualifier for CICS IA dump data set that are produced.

13 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 data is

collected. If you are aware that your environment or your application does not perform dynamic program calls, specify N (No). The “Y” value instructs the collector to use a special CICS TS XPI, IDENTIFY_PROGRAM, to identify the program name. If a detected command belongs to a program that is not known to CICS TS, CICS TS returns a name of the top-level program instead. In this case, the CICS IA collector records a top-level program name and marks the record with '-1' (X'FFFFFFF') as a command offset.

14 Trigger for Task collection

This field is only for dependency collection. The default value is 1. The value is used to reduce the CICS IA Dependency collector performance overhead on the CICS region. When the **Trigger for Task collection** field is 1, all CICS tasks are permitted for collection, the CICS tasks still might not be collected because of other options, including the program exclude list. If you assume that the **Trigger for Task collection** is N and greater than 1, only every Nth task is permitted for collection, beginning from the first CICS task start after the start of CICS IA Dependency collector. The transaction exclude list is applied to CICS tasks before this option is applied.

Note:

This option is task associated. When a CICS task starts, further changes of this option and collector restart does not take effect on collection rules for this task.

If these conditions are true:

- The **Collect Long Running Tasks** field is set to N.
- The **Trigger for Task collection** field is greater than 1.
- The Dependency collector is already running.

The long running tasks that were already started before the Dependency collector are not collected. If the **Collect Long Running Tasks** field is set to Y then the long running tasks that are started after the Dependency collector are collected.

15 Collect Long Running Tasks

This field is only for dependency collection and can be used to enable the collection of all long running CICS tasks, disregarding the value set in the **Trigger for Task collection** field. The default value of this option is N.

When the current setting of the **Trigger for Task collection** field is 1, this option does not affect the data collection because all Long Running CICS Tasks are already permitted for collection.

When the current setting of **Trigger for Task collection** field is greater than 1, and this option is set to N, all CICS Tasks that are not permitted for collection by the **Trigger for Task collection** option, are never collected.

When the current setting of the **Trigger for Task collection** field is greater than 1, and this option is set to Y, the following CICS Tasks become collection-enabled:

1. CICS Tasks that were started when the Dependency Collector was inactive.
2. CICS Tasks that were started after the Dependency Collector was started and the **Trigger for Task collection** was set to 1.

3. CICS Tasks that were started after the Dependency Collector was started and that were not permitted for collection by the **Trigger for Task collection** option, when the runtime of these CICS Tasks is more than 30 minutes.

16 Help

Pressing F1 for help does not save any changes that are 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.

The following 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 Collector Resource Options panel, CIU290. Press Enter.
The CICS Resources Options panel, CIU240, shown in Figure 23 on page 114, 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 Collector Resource Options panel, CIU290. Press Enter.
The CICS Resources Options panel, CIU245, shown in Figure 24 on page 115, 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 - V5R3M0

2014/08/25

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 **1**
 D=Yes+Detail, (Only for API types marked with *) **3**

2 APIs

*Programs . . . _

*Files _

*Transactions . _

Task Control . _

Presentation . _

*TS Queues . . _

*TD Queues . . _

Journals . . . _

DTP _

Counters . . . _

FEPI _

*WEB Services . _

*Exits _

Others _

*EVENTS _

ATOMServices . _

XMLtransform . _

WSAddressing . _

CICS Sysid: _____

CICS Applid: _____

TermID: _____

F1=

F2=

F3=Exit

F4=

F5=

F6=

F7=

F8=

F9=

F10=

F11=

F12=Cancel

Figure 23. Collector CICS Resources Options panel, CIU240

- 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 “CICS API commands” on page 217. The individual EXEC CICS FEPI commands that make up the FEPI API group are listed in “CICS FEPI API commands” on page 236.
- 3 D=Yes+Detail option**

You can use option D only for the API types marked with an asterisk.

If you are going to run the threadsafe report you must collect Detailed information for Programs and Files, select **D** for **Programs** and **Files**.

CIU245	CICS Interdependency Analyzer for z/OS - V5R3M0	2014/10/21
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 1		
D=Yes+Detail, (Only for SPI types marked with *) 3		
SPIs (Create/Inquire/Set/Discard/Perform) 2		
Programs . . . _	Files _	Transactions . _
Transient Data _	DB2 _	DJAR _
Corbaserver . _	TCPIPService . _	FEPI _
Library. . . . _	*Connections . _	BTS preoc . . . _
ATOMServices . _	CSD. _	XMLTransform . _
JVMServer. . . _	Terminals. . . _	CICS System . . _
DUMPs _	VTAM Conn. . . _	Statistics . . . _
SHUTDOWN . . . _		Temp Storage . _
		BRFacility . . . _
		Journals _
		Bundles. _
		MQCONN _
		Tasks _
		Tracing _
CICS Sysid: _____	CICS Applid: _____	TermID: _____
F1=	F2=	F3=Exit
F7=	F8=	F9=
		F4=
		F10=
		F5=
		F11=
		F6=
		F12=Cancel

Figure 24. Collector CICS Resources Options panel, CIU245

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 225. The individual EXEC CICS FEPI commands that make up the FEPI SPI group are listed in “CICS FEPI SPI commands” on page 236.

3 D=Yes+Detail option

You can use option D only for the SPI types marked with an asterisk.

For more information about restrictions affecting the monitoring of commands that might cause dependencies or affinities, see “What the Collector does not monitor” on page 17 and Appendix A, “Details of dependencies and affinities collected,” on page 217.

Specifying region-specific options: Application data collection

You can configure the collection of CICS TS Application data associated with programs and other CICS TS Resources from the Application Data Collection Options screen.

1. If you specified **ALL** on the Region Configuration menu the initial value of the **Enable collection of Application Data** field on the Application Data Collection Options screen is set to the first CICS region listed on the Region Configuration menu. That is, the value of the first region listed after the special ALLand

DEFAULTAPPLIDs. Both the CICS APPLID and the SYSID are shown as ALL. Selected Application is not displayed in this mode.

2. If you specified **DEFAULTS** on the Region Configuration menu the initial values of the fields on the Application Data Collection 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 updates 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**, you cannot leave a blank.

To call up this screen, from the Resource Options screen, CIU290, type action code 8, and press Enter. The Application Data Collection Options screen, CIU210, is displayed:

CIU210

CICS Interdependency Analyzer for z/OS - V5R1M0
Application Data Collection Options

2012/08/30
03:28:35PM

Type Collection Option then press ENTER :

Enable collection of Application Data: Y

1

Selected Application:

2

NAME: CICS_TS_SAMPLE_APPLICATION
VERSION: 2.4.1

CICS Sysid: T51B CICS Applid: IYDZT51B TermID: TC10

F1=Help F2= F3=Save&Exit F4= F5= Refresh F6=

F7= F8= F9= F10= F11= F12=Cancel

Figure 25. Application Data Collection Options screen, CIU210

Note:

1 Enable collection of Application Data

- Enter Y (Yes) to capture application associated data within your CICS region or regions.
- If you are editing the DEFAULTS record, you must specify Y or N. You cannot specify a blank.
- If this option is N (NO), selected application is still displayed, but Application Data is not collected.

2 Selected Application

- Displays Application selected in CICS IA plug-in for CICS Explorer. If no Applications are selected the value is set to ALL.
- You cannot change the selected application from this screen.

Specifying which dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored

Use the Resource Options panel CIU290, to specify which dependency-related commands are monitored.

To specify whether dependency-related CICSplex SM, DB2, IMS, and MQ commands are to be monitored on a particular CICS region, type action code 5 at the top of the Resource Options panel CIU290, as shown in Collector Resource Options panel, CIU290, and press Enter.

The DB2/MQ/IMS/CPSM Resource Options panel, as shown in Figure 26, is displayed.

Note:

1. If you specified **ALL** on the Region Configuration Menu, the initial values of the fields on the DB2/MQ/IMS/CPSM Resource 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/CPSM Resource 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 edit the DEFAULTS record, you must specify **Y** or **N** in each field. You cannot specify a blank.

CIU250
CICS Interdependency Analyzer for z/OS - V5R3M0
2014/06/24

DB2/MQ/IMS/CPSM
Resource Options for
02:31:14PM

CICS Sysid : Z518
CICS Applid : IYDZZ518

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

Detect command types: Y=Yes, N=No or blank=default **1**

DB2 Options

Collect DB2 Resources : Y
Collect resource name : N **2** (Y/N) (Access the SYSIBM.SYSPACKSTMT and the SYSIBM.SYSSMT tables)

MQ Options

Collect MQ Resources : Y

IMS Options

Collect IMS Resources : N

CPSM Options

Collect CPSM Resources : N

CICS Sysid: Z518
CICS Applid: IYDZZ518
TermID: TC45

F1=
F2=
F3=Exit
F4=
F5=
F6=

F7=
F8=
F9=
F10=
F11=
F12=Cancel

Figure 26. Collector DB2/MQ/IMS/CPSM 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 CICSplex SM, DB2, MQ, and IMS groups are listed in “Non-CICS API commands detected” on page 237. 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 the Collector does not monitor” on page 17.

2 DB2 Options: Collect resource name

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**.

Specifying which affinity-related CICS commands are to be monitored

Use the Resource Options panel, CIU290, and the CICS Affinities Options panel, CIU270, 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 Collector Resource Options panel, CIU290, and press Enter.

The CICS Affinities Options panel, CIU270, shown in Figure 27 on page 119, 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 - V5R1M0      2015/05/06
          CICS Affinities Options for                      03:26:22PM
          CICS Sysid   : Z53B   CICS Applid   : IYDZZ53B

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

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

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

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

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

CICS Sysid:  Z53B   CICS Applid:  IYDZZ53B   TermID:  TC68

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

```

Figure 27. Collector Affinities Options panel, CIU270

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.

You can choose the symbol T for TS QUEUE type, if you want the Collector to monitor only terminal associated tasks.

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.

4 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 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 the option 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.

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 99 and “Resuming the collection of data” on page 100. 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 Collector Region Configuration Menu panel, CIU200. 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 28 on page 121, 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 - V5R1M0	2012/08/09
	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 12 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
F7=	F8=	F9=
		F10=
		F11=
		F12=Cancel

Figure 28. Collector Time and Dates Options screen, CIU280

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.

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 6, and press Enter. The Natural Options screen, CIU29N, looks as follows:

```

CIU29N          CICS Interdependency Analyzer for z/OS - V5R1M0      2012/08/25
                  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 or blank=default

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 29. Collector Natural Options screen, CIU29N

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 Collector Main Administration Menu panel, CIU000). The Collector Global Options Menu panel, Figure 30, 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 the CICS IA collector is terminated when a CICS IAerror occurs
- 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

CIU300

CICS Interdependency Analyzer for z/OS - V5R1M0

2012/08/19

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)

Terminate on Abend : N (Yes/No)

High Level Trace : N (1/2/3/No)

Restore Trace flags. : N (Yes/No)

National Language Option : E Code: ENU

Date and Time Formats

Date 4 1. MMDDYY 2. DDMMYY Separator /

3. YYMMDD 4. YYYYMMDD

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

CICS Sysid: TLS3 CICS Applid: IYCLZC03 TermID: TC20

F1=Help F2= F3=End F4= F5=Refresh F6=

F7= F8= F9= F10= F11= F12=Cancel

Figure 30. Collector Global Options Menu panel, CIU300

Note:

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 Terminate on Abend

This option is used to specify whether the Collector is terminated with an IUZT abend when an error occurs when CICS IA queries DB2. Type Y or N.

3 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 at the entry and exit of a module.
- The points of level 2 are included in CICS IA modules in all other cases.
- The points of level 3 are included in GLUE or TRUE.

You can set this levels through the CINT transaction.

4 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.

5 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. |

6 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 “:”.

Managing the Command Flow collection using transaction CINC

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.
- Configure the initial part of the Command Flow data collection options.
- Show information about the current state of the CINC collector.
- 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.
- Display the CICS IA Command Flow Application Data Collection panel, CIUA0A, to view or change the Application Data collection options.

Procedure

1. At a CICS terminal, type the transaction identifier CINC.
2. Press Enter. Panel CIUA01 is displayed; see Figure 31 on page 126.

CIUA01	CICS IA Command Flow Options	1	ApplID	IYDZT51B
Command Flow state	STOPPED	2		
Date/Time of last start.	2014/03/22 03:15:34PM	3		
Date/Time of last stop	2014/03/22 03:15:48PM	4		
Command Flow Id	MYCMDID	5		
Traced user USERIDs	CICSUSER	6	CINC Authority: GENERAL	8
Traced terminal TermIDs.	*	7		
Traced transaction IDs	TWEB TR* T+AN TR%N T++N		(Max 5 transact. IDs)	9
Exclude lists	Transaction		Program	10
User Journal Name.	CIUMTJNL	11		
Journal Copy Criteria	LAST		(LAST, USER or CFID)	12
User Exit Name.				13
Dynamic Call	Y		(Yes/No)	14
Tasks before Stopping	0		(0-9999)	15
Records before Stopping.	0		(0-9999999)	16
CIU7000I 5655-Y22 (C) Copyright IBM Corp. 2001,2014				
F1=Help	F2=	F3=Exit	F4=Options	F5=Start
F7=Stats	F8=	F9=	F10=Applications	F11=
				F6=Stop 17
				F12=Cancel

Figure 31. 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

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 ID associated with CICS transactions, for which you want to collect the Command Flow data.

Note: A PRIVILEGED CINC User can use %, +, and Wildcard *. Wildcard * can be used only as a suffix.

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

8 The CINC user can be defined as a GENERAL user or a PRIVILEGED user. PRIVILEGED users can use wildcards to collect data for more than one user ID. The default value is General.

9 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.

10 The names for program and transaction exclude lists that are to be used by command flow collector. The exclude list format is the same with Dependency/Affinity collector (CINT). If the transaction name in the list of collected transactions does not contain wildcards, the transaction exclude list will not be applied to it.

11 The journal name for trace data. The length of the name can be up to 8 characters. The default journal name is CIUMTJNL.

12 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.

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

14 Capturing the dynamic calls. Specify Y (Yes) or not N (No). The default value of this field is Y. The Y value instructs the IA collector to use a special CICS TS XPI, IDENTIFY_PROGRAM, to identify the program name. If a command is detected and that command belongs to a program that is not known to CICS TS, CICS TS returns the name of the top-level program instead. In this case, IA collector records the top-level program name and marks the record with -1 (X'FFFFFFFF') as a command offset.

15 The number of CICS tasks processed before stopping the collector. The default value of this field is 0. This value is used to indicate that there is no limit to the number of tasks for which command flow data is collected.

16 The number of CICS IA journal records collected before stopping. The default value of this field is 0. This value is used to indicate that there is no limit to the number of journal records collected.

17 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 11.

Table 11. 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
Accessing Application Data collection options in order to view or change them. The CICS IA Command Flow Application Data Collection panel, CIUA0A, is displayed. Note: This function does not preserve the Command Flow options changes made on the CIUA01 panel.	F10

Note:

- 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.

- c. Natural programs control flow is collected if SYSRDC exit is active and scheduled at least once. It is recommended to use CIUXPROG exclude list to exclude unnecessary entries from the Command Flow collection.
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 32.

CIUA02
CICS IA Command Flow ApplID list **1**
AppID IYDZZ41A

Press F4 for select the region OR type regions and press ENTER

CICS ApplIDs. : IYDZZ41A _____ **2**
: _____
: _____
: _____

F1=Help	F2=	F3=Exit	F4= Prompt for Regions	F5= Delete	F6= 3
F7=	F8=	F9=	F10=	F11=	F12=Cancel

Figure 32. 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 12.

Table 12. 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
Returning to the CICS IA Command Flow Options panel (CIUA01) without saving any changed Command Flow options.	F12
Getting the list of available CICS regions (Prompt for Regions).	F4
Delete the selected region from the ApplIDs list (Delete).	F5

Note: With the CICS IA Command Flow ApplID list panel you can set up your own regions configuration. This panel contains the list of CICS APPLIDs (up to 15 regions) that are to be monitored by the Command Flow Utility. You can form this list by using the **Prompt for Regions** function (F4 key), and the Delete function (F5 key).

- Press F3 (or F12) to close the CICS IA Command Flow ApplID list panel and return to the CICS IA Command Flow Options panel, CIUA01. Closing the panel does not affect the state of the CINC collector.

Displaying the CICS IA Command Flow Application data collection panel

With the Command Flow Application data collection panel you can configure collection of CICS TS Application data associated with programs and other CICS TS Resources.

About this task

You can use the Command Flow Application data collection panel to switch the Application Data collection feature of CICS IA Command Flow Collector ON and OFF. This screen also displays application selected for collection in CICS IA plug-in for CICS Explorer

Press F10 (Applications) on the Command Flow Application data collection panel, CIUA0A is displayed.

CIUA0A
CICS IA Command Flow Application Data Collection
ApplID IYDZT51B

Type Collection Option then press ENTER :

Enable collection of Application Data: Y
1

Selected Application:
2

ALL.

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

Figure 33. Command Flow Application Data Collection panel, CIUA0A

Note:

1 Enable collection of Application Data

- Enter Y (Yes) to capture application associated data within your CICS region(s).
- If this option is N (NO), the selected application is still displayed but the Application Data is not collected.

2 Selected Application

- Displays Application selected in IA plug-in for CICS Explorer. If no Applications selected, its value will be set to ALL.
- It is not possible to change selected application from this screen.

Displaying CICS IA Command Flow Statistics panel

With the CICS IA Command Flow Statistics panel you can monitor all the regions that you have specified in your own regions configuration.

About this task

You can use the CICS IA Command Flow Statistics panel to display statistics for all of the specified regions from your regions configuration.

Procedure

1. Press F7 (Statistics) on the CICS IA Command Flow Options panel, CIUA01. The CICS IA Command Flow Statistics panel, CIUA03, is displayed. See Figure 34 on page 132.

CIUA03

CICS IA Command Flow Statistics

1 ApplID IYDZZ41A

CICS ApplID

CICS SysID

Records written

Collector/Region state

2

IYDZZ41A

Z41A

0

STOPPED

CIU2324W CINC Collector is not started in all your regions

F1=Help

F2=

F3=Exit

F4=

F5=

F6=

F7=

F8=

F9=

F10=

F11=

F12=Cancel

3

Figure 34. CICS IA Command Flow Statistics panel, CIUA03

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 and CICS SysID columns 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 13.

Table 13. 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.

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 35.

Figure 35. List of available CICS regions , CIUA04

1 The 8-character APPLID of the CICS region on which the CICS IA Command Flow Utility is running.

3 The CICS APPLID fields column. All the regions from this list are defined in the CINT control file.

Table 14. The User Command Flow Utility control keys on the CIUA04 panel

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.

Operating CICS IA Collection from the CICS IA Explorer plug-in

In CICS IA V5.3, you can manage data collection not only by using the CINC and CINT transactions, but also by using the CICS IA Explorer plug-in.

The CINT transaction configures CICS IA to manage multiple regions.

1. Share the CIUCNTL control file.
2. Define the CONNECTION and SESSION resources between the controlling CICS region and the regions you want to manage.

In CICS IA V5.3, you can use the CICS IA Explorer plug-in to

- Display the status of regions, users, and collection processes.
- Operate the dependency and affinity data collections. You can START, STOP, PAUSE, and CONTINUE the collection process.
- Set up application options for the dependency and the affinity collections.
- Operate the Command Flow collector (START and STOP) for the logged in user.
- Set up the Command Flow options for the logged in user.
- Reset the collection parameters.

Also, the CICS IA Explorer V5.3 plug-in introduces the new **Scenario based collection** feature. Now, you can set up the required options and start a **Threadsafe** or **Affinity** collection scenario with one-click on a selected region in the **IA Operations** view. Additional options: **DB2**, **IMS**, **MQ**, **CPSM**, **NATURAL/ADABAS** and **Optimum performance** are also available.

To administer the data collection tasks by using the CICS Explorer plug-in, you must define web service resources in the controlling region. For more information, see “Controlling CICS IA from the CICS IA plug-in” on page 48.

Chapter 5. 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.

CIUUPDB1

Updates the CPSM and 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 WebSphere MQ table, CIU_MQ_DATA, from the WebSphere 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, and CICSplex SM, DB2, MQ, Natural, and IMS, from all the dependency data files, CIUINT1 through CIUINT7.

Note:

If you are not collecting DB2 data, you can remove step STEP060 through STEP091.

If you are not collecting MQ data, you can remove step STEP100 through STEP131.

If you are not collecting IMS data, you can remove step STEP140 through STEP171.

If you are not collecting Natural and Adabas data, you can remove step STEPN10 through STEPN41.

Note: In sample jobs CIUUPDB and CIUUPDB1, which are used to load data into the CIU_CICS_DATA table, an optional step is introduced with CICS IA V5.3. This step is called STEP048 and is used to reload the CIU_CICS_CHAINP table by collection ID. STEP048 calls a DB2 stored procedure, which is available only if your CICS IA database is running on DB2 V9.1, or later.

The CIU_CICS_CHAINP table is used by the new DB2 views and stored procedures, which are used to identify “Application” entry points when you want to deploy the new “Platform and Application” features in CICS Transaction Server V5, and later. If you are not planning to use this function, then you can comment out this step.

CIURESID

Updates the CIU_RESOURCE table that is used by the CICS IA plug-in for CICS Explorer.

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, however, you have separate sets of dependency data files for each region, you can run the jobs one time 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. After you record all of your applications, you might need to update the database only for new applications and changes.

CICS IA V3.2 introduced the ability to identify a CICS dependency collection at DB2 load time. This ability enables the user to load, manage, and compare resource usage by collection ID. Before you submit the job, you must review the collection name that you use. The job is first setup with a collection ID of `_collid_`. This collection ID is used in several places so you must run a global change to the collection ID of your choice. The collection ID can be up to 16 characters in length. If you want to change the collection ID every time you run the update jobs, you can edit the job, run "FIND COLLID=", and replace the collection ID name appropriately.

For more information about how to view and manage your data by collection ID, see Chapter 6, “Managing your CICS IA data,” on page 139.

Database update procedure

The job streams to update the dependency tables are put into the hlq.SCIUSAMP.CICS library by the CICS IA installation procedure, where “hlq” is a prefix that is defined during installation.

Before you run any of the jobs, 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.SCIUSAMP.CICS 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 data set.

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 data set.

Chapter 6. Managing your CICS IA data

With each new release CICS IA captures more data. In volatile development environments this data can soon become out of date. If you have out of date data, the CICS IA plug-in might retrieve more data than you need. The presence of unwanted data might impede your analysis and provide confusing results.

The data can be stored in three places at the same time:

- The CICS IA data space when the collector is running
- The CICS IA VSAM collector VSAM files
- The CICS IA database

This information describes how to manage the data you collect, and manage the data that you have collected already.

Identifying the data that you collect

You can reduce the amount of data you initially collect by identifying patterns in your resource names, and by using a collection ID to identify the data you load into DB2.

Identifying patterns in resource names

Some CICS resource types can have resource names that are based on a prefix or a suffix concatenated with a counter or an address. These resource names can create millions of records that are stored in the CICS IA data space, VSAM files, and the DB2 database.

For example, if you could have a CICS TSQUEUE resource with a name based on `QUEUEn`, where: *n* is a 10-digit string that is controlled by a CICS counter.

If you have `WRITEQ`, `READQ` and, `DELETEQ` commands for this TSQUEUE resource in the program, CICS IA, will have three entries for each TSQUEUE name that is processed, starting with `QUEUE0000000001` and ending with `QUEUE9999999999`. Based on this format, there are already 3 billion records for that single program. The number of records can increase substantially if the data for that program is collected in more than one region, or if the queue name is used by more than one program.

Because of this issue, CICS IA uses a function to reduce the number of records as described in this scenario to only three by replacing the numerics with `+++++`. These records are:

- `WRITEQ QUEUE+++++`
- `READQ QUEUE+++++`
- `DELETEQ QUEUE+++++`

You can reduce the number of records to three if you create a “resource compression list” table. This resource compression list consists of a simple assembler CSECT where you can use prefixes, suffixes, or any pattern. Currently you can create these lists for CICS TSQUEUE names, CICS CHANNEL/CONTAINER names, CICS ENQ/DEQ names, and WebSphere MQ queue names. You can configure the same table for each CICS region or you can configure individual ones dependent upon the content of that region.

When you first start collecting CICS IA data, use the CICS IA plug-in for CICS Explorer to identify the resource names that match a pattern, and add these names to your resource compression list. You must remove the duplicated data by deleting them from the VSAM file and the DB2 tables. A clean up job is provided to remove duplicate resource names for CICS TS QUEUE names, CICS ENQUEUE, CICS DEQUEUE, and WebSphere MQ queue names.

For more information about removing duplicate resources, see “Removing duplicate resources from the Collector files” on page 147.

For more information about resource compression lists, see “Creating a resource compression list” on page 77.

Identifying the data by using a collection ID

The COLLECTION_ID column in the CICS Dependency Data tables is assigned to the data during the DB2 load utilities for Dependency data.

Why use a collection ID?

If your CICS development or test environment, or both, are volatile, they are subject to frequent program changes, if you use a different collection ID every time you load the data it can help you to analyze the data in the CICS IA plug-in and, eventually, to delete unwanted collections. If you load the data on a weekly basis, it is beneficial to identify the DB2 load with a unique weekly identifier. If you have a unique weekly identifier, you can:

- Reduce the amount of data that you retrieve in the CICS IA plug-in if you set the scope to be the collection ID.
- Compare data in the CICS IA plug-in by collection ID.
- Delete the data for the unwanted collection IDs.

For more information about how to set the scope for a collection ID, see the online help shipped with the CICS IA plug-in.

You can also use the COLLECTION_ID column when you upgrade your CICS environment, or embark on a major project where you will use CICS IA data from before and after the upgrade. You can give the data that is captured during testing before you upgrade a collection ID name such as “before image” and, similarly, after the upgrade you can use a collection ID name such as “after image”. This naming convention means you can analyze the data in the CICS IA plug-in more easily, and delete the “before image” data when the upgrade is complete.

Managing the collected data

To manage the data that you have collected, you can delete your unwanted data and keep your data sources in sync.

Utilities that are provided by CICS IA to delete data from the DB2 database. These utilities include the following jobs:

- Sample job SCIUSAMP.DB2(CIUCLR). This job provides the skeleton JCL and SQL to delete data that is based on specified conditions from the CICS IA tables.
- Sample job SCIUSAMP.CICS(CIUPRUNE). This job provides JCL and SQL to delete data that is based on the last used date from the tables.
- Sample job SCIUSAMP.CICS(CIUDELSP). This sample job calls the DB2 stored procedure to delete data that is based on the collection ID from all of the tables.

- Sample job SCIUSAMP.CICS(CIUPURSP). This sample job calls the DB2 stored procedure to delete data that is based on the prefix of the object name from CIU_CICS_DATA table.
- Sample jobs SCIUSAMP.CICS(CIULPGSP) and SCIUSAMP.CICS(CIUDPGSP). These sample jobs call the DB2 stored procedure to list (CIULPGSP) and delete (CIUDPGSP) data that is based on the **collection_id**, **applid**, and **program_name** values taken from the specified table.
- If you use the CICS IA plug-in for CICS Explorer you can delete data by TS application and by collection ID.

Identifying data to delete

The CICS IA data that you collect can change, which can lead to redundant records being stored in the database. You can delete data from the database in a number of ways. You can use SQL queries to help you to delete the unwanted data from your database tables, which improves performance.

The CICS IA data is loaded into the DB2 tables from the VSAM files. You must ensure that you clear the VSAM files and the DB2 tables. For development and test regions, it is prudent to empty the files and tables at CICS IA startup.

The supplied a sample job, h1q.SCIUSAMP.CICS(CIUPRUNE), is used to delete data from the DB2 tables. You can use this job to run sample supplied SQL that is provided in h1q.SCIUSQL.CICS. These sample SQL jobs begin with the prefix CIUP; for example, CIUPCICS deletes data from the CIU_CICS_DATA table. The sample jobs use the LAST_RUN date. The query first displays the records before the selected LAST_RUN date:

```
--
-- Show rows that are older then specified timestamp
SELECT APPLID, TRANSID, PROGRAM, FUNCTION, TYPE, OBJECT
FROM CIU_CICS_DATA READONLY
WHERE LAST_RUN<='2012-01-01-00.00.00.000000';
-- Uncomment this statement when you want to delete rows
--DELETE FROM CIU_CICS_DATA
--
-- WHERE LAST_RUN<='2012-01-01-00.00.00.000000';
COMMIT;
```

Figure 36. Records before the selected LAST_RUN date.

An alternative method involves deleting records based on the program version. You do this by querying on the program, the program length, and the first used timestamp. Program and program length are part of the unique key that is used the data is stored in the CICS IA dependency tables.

When a program changes, the program length probably changes too. CICS IA uses this information as a crude method for a programming version. The EXEC commands for a changed program contain a FIRST_RUN timestamp, which is used to identify the latest program. You can then delete all of the commands that are reported for previous programs.

If you want to use the method that involves deleting records based on the program version, you must first examine the data to look for programs with different program lengths. In the following figure, program DB900001 has two different program lengths. From the FIRST_RUN timestamp, you can see that program length 2180 corresponds to the latest program.

```

SELECT DISTINCT APPLID, PROGRAM, PROGLen,
MAX(CAST(FIRST_RUN AS CHAR(26)))
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM , PROGLen;
-----+-----+-----+-----+-----+-----+-----+-----+-----+
APPLID PROGRAM PROGLen
-----+-----+-----+-----+-----+-----+-----+-----+-----+
IYCYZC37 DB900001 000021E8 2007-10-30-11.31.33.000000
IYCYZC37 DB900001 00002180 2007-11-22-11.23.57.000000
IYCYZC37 DB910001 000032A0 2007-11-22-11.23.57.000000
IYCYZC37 DB910001 000033E0 2007-10-30-11.31.33.000000
IYCYZC37 DB930001 00002000 2007-11-22-11.23.57.000000
IYCYZC37 DB930001 00002050 2007-10-30-11.31.33.000000
IYCYZC37 DB940001 00002698 2007-11-22-11.23.57.000000
IYCYZC37 DB940001 00002720 2007-10-30-11.31.33.000000
IYCYZC37 EMSTESTA 00002110 2008-04-03-12.38.48.000000
IYCYZC37 EMSTESTB 00001500 2008-04-03-12.38.48.000000

```

Figure 37. Query to show program versions and FIRST_RUN timestamps.

You can now find all of the latest programs in region IYCYZC37. The query in the following figure shows the latest FIRST_RUN timestamp. In this query, the output is concatenated so it can be used later as an SQL subquery.

```

SELECT DISTINCT APPLID||PROGRAM||MAX(CAST(FIRST_RUN AS CHAR(26)))
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM ;
-----+-----+-----+-----+-----+-----+-----+-----+-----+
-----+-----+-----+-----+-----+-----+-----+-----+-----+
IYCYZC37DB9000012007-11-22-11.23.57.000000
IYCYZC37DB9100012007-11-22-11.23.57.000000
IYCYZC37DB9300012007-11-22-11.23.57.000000
IYCYZC37DB9400012007-11-22-11.23.57.000000
IYCYZC37EMSTESTA2008-04-03-12.38.48.000000
IYCYZC37EMSTESTB2008-04-03-12.38.48.000000
IYCYZC37EMSTESTS2008-04-03-12.38.45.000000
IYCYZC37FC0100012007-11-19-15.09.18.000000
IYCYZC37FC0200012007-11-19-15.09.17.000000

```

Figure 38. Query to show all distinct programs by latest FIRST_RUN timestamp.

You can now get the program length for the latest programs. The query in the following figure shows the program and program length for the most recently used programs. The output is concatenated so it can be used later as an SQL subquery.

```

SELECT DISTINCT APPLID||PROGRAM||PROGLEN
FROM CIU_CICS_DATA ,(
SELECT DISTINCT APPLID AS A, PROGRAM AS P,
MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM) X
WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S;
-----+-----+-----+-----+
-----+-----+-----+-----+
IYCYZC37DB90000100002180
IYCYZC37DB910001000032A0
IYCYZC37DB93000100002000
IYCYZC37DB94000100002698
IYCYZC37EMSTESTA00002110
IYCYZC37EMSTESTB00001500
IYCYZC37EMSTESTS00003380
IYCYZC37FC01000100000AD0
IYCYZC37FC02000100001558

```

You can now display the records for all of the commands that are collected from earlier versions of the programs, which are displayed in the query in Figure 5. You can now choose to delete these records, as shown in Figure 6.

```

SELECT * FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
AND APPLID||PROGRAM||PROGLEN NOT IN
(SELECT DISTINCT APPLID||PROGRAM||PROGLEN
FROM CIU_CICS_DATA , (
SELECT DISTINCT APPLID AS A, PROGRAM AS P,
MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM) X
WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S)
-----+-----+-----+-----+-----+-----+
APPLID HOMESYSID TRANSID PROGRAM FUNCTION TYPE OBJECT
-----+-----+-----+-----+-----+-----+
IYCYZC37 TS07 DB90 DB900001 RECEIVE MAP S1
IYCYZC37 TS07 DB90 DB900001 RECV MAP MAPSET CBMS01
IYCYZC37 TS07 DB90 DB900001 SEND MAP S1
IYCYZC37 TS07 DB90 DB900001 SEND MAP S1
IYCYZC37 TS07 DB90 DB900001 SEND MAP MAPSET CBMS01
IYCYZC37 TS07 DB90 DB900001 SEND MAP MAPSET CBMS01
IYCYZC37 TS07 DB91 DB910001 RECEIVE MAP S2

```

You must repeat this delete action for the other tables: CIU_DB2_DATA, CIU_IMS_DATA, and CIU_MQ_DATA.

```

-----+-----+-----+-----+-----+-----+
DELETE FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
AND APPLID||PROGRAM||PROGLEN NOT IN
(SELECT DISTINCT APPLID||PROGRAM||PROGLEN
FROM CIU_CICS_DATA , (
SELECT DISTINCT APPLID AS A, PROGRAM AS P,
MAX(CAST(FIRST_RUN AS CHAR(26))) AS S
FROM CIU_CICS_DATA
WHERE APPLID='IYCYZC37'
GROUP BY APPLID, PROGRAM) X
WHERE CAST(FIRST_RUN AS CHAR(26)) = X.S)
-----+-----+-----+-----+-----+
DSNE615I NUMBER OF ROWS AFFECTED IS 24
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+

```

Figure 41. Query to delete commands from the CICS table for old commands.

You can also use these queries as the basis for writing your own delete functions.

When the data from these tables is deleted, you must reload the CIU_RESOURCE table that is used by the CICS IA plug-in by running the sample job SCIUSAMP.CICS(CIURES LD).

Deleting by collection ID

You can now delete CICS IA data from all the dependency tables by using the stored procedure CIUSPDEL, and using a collection ID.

You can call the stored procedure CIUSPDEL from the sample job SCIUSAMP.CICS(CIUDELSP) or from the CICS IA plug-in.

To delete the data by using the CICS IA plug-in, right-click the name of the collection ID for which you want to delete data in the Collection IDs window, then click **Delete associated data**. The stored procedure CIUSPDEL deletes the data, based on the collection ID, from all the CICS dependency tables.

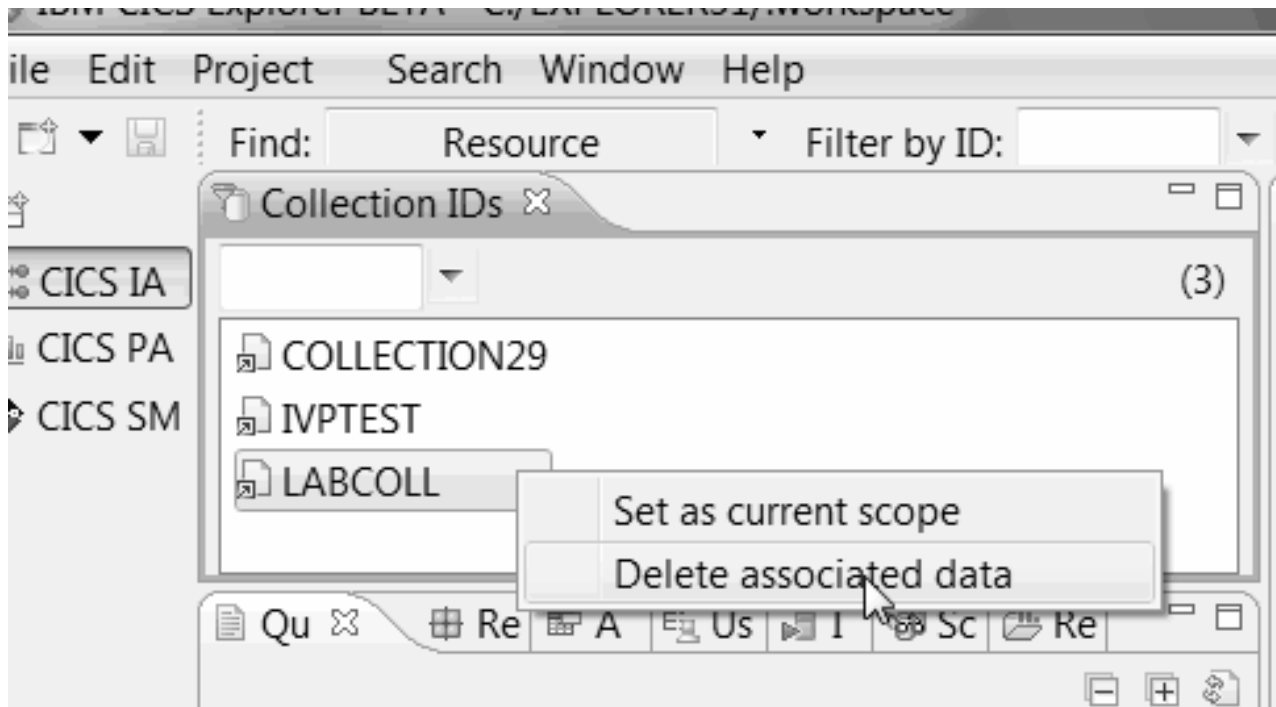


Figure 42. The CICS IA plug-in Collection IDs window

Deleting TSQ or ENQ – related resources with a duplicate name prefix

You can now delete CICS IA data from some dependency tables by using the stored procedure CIUSPPUR, and using a prefix of the name for TSQ or ENQ objects. This resource names should consists of a constant prefix and a variable parts.

You can call the stored procedure CIUSPPUR from the sample job SCIUSAMP.CICS(CIUPURSP) with the two parameters, **type** and **prefix**. The result is a large amount of unnecessary TSQ or ENQ related dependency records for the same combination of type, function, program, offset, proglen and prefix of resource is compressed and the size of the CIU_CICS_DATA and the CIU_RESOURCE tables is optimized. For more information, see “CIUSPPUR Stored Procedure” on page 425.

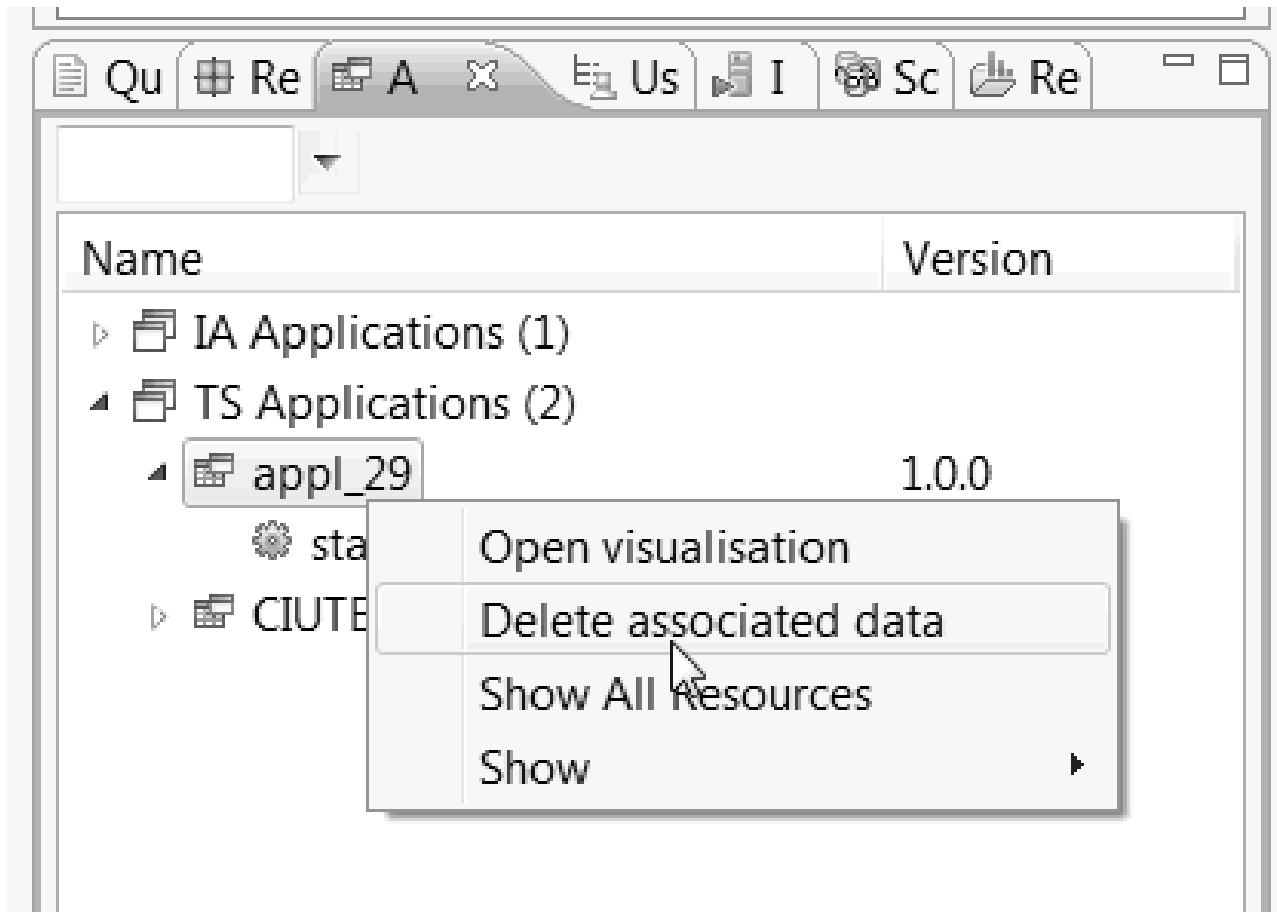
Deleting by application ID

In CICS Transaction Server you can deploy applications in a platform. You can define the various resources that make up a business application in CICS as a single entity and deploy these resources to CICS as a single resource.

An application that is defined as part of the platform can be managed as a single entity throughout the lifecycle, making CICS application management faster, easier, and less prone to error. For more information about applications, see Applications in the CICS TS 5.2 Knowledge Center.

The CICS IA exits capture application information and associate them with the dependency data and the command flow data that is collected.

You can use the CICS IA plug-in to delete the data by application and version. Right-click the name of the application for which you want to delete data, then click **Delete associated data**.



Deleting old versions of programs

You can delete CICS IA data from different dependency tables. The tables might contain many records that relate to old versions of a program. By deleting the unwanted records, you reduce the size of your tables and can improve the efficiency of your system.

You can use the CIUSPDGP DB2 stored procedure to delete these redundant records by specifying the COLLECTION_ID, APPLID, and PROGRAM NAME.

To identify the redundant data, you can call the stored procedure CIUSPDGP from the sample job SCIUSAMP.CICS(CIULPGSP) and enter the values for the following parameters:

- **collection_id**
- **applid**
- **program_name**
- **table_name**

This sample job contains a LIST option for the stored procedure and the job populates the CIU_PROGRAM_TEMP1 and CIU_PROGRAM_TEMP2 tables.

The CIU_PROGRAM_TEMP1 table contains the latest versions of the specified programs. These rows are to be kept and are not deleted.

The CIU_PROGRAM_TEMP2 table contains the rows from the specified table, which can be deleted.

The CIU_PROGRAM_TEMP1 and CIU_PROGRAM_TEMP2 tables contain the information only from records, which are kept or deleted.

Some of the records might contain the same information, so, the count of the number of kept or deleted records can differ from the number of records in the CIU_PROGRAM_TEMP1 and CIU_PROGRAM_TEMP2 tables.

After the job is finished, you can review the CIU_PROGRAM_TEMP1 and CIU_PROGRAM_TEMP2 tables. Ensure that the content in the CIU_PROGRAM_TEMP2 table is no longer required.

You can then run the SCIUSAMP.CICS(CIUDPGSP) sample job, which contains a DELETE option. The CIUSPDPG DB2 stored procedure deletes all of the rows that are listed in the CIU_PROGRAM_TEMP2 from the specified table.

Attention: You must run the CIUSPDPG stored procedure with the LIST option before you can delete the data. Otherwise, the deletion job does not work. For more information, see “CIUSPDPG Stored Procedure” on page 416.

Managing affinity data

You can capture affinity data by CICS region, and you can use the affinity data when you want to implement workload balancing across cloned regions.

Capturing affinity data is typically a one-off task and it is recommended that you empty both the affinity files and the DB2 tables that are associated with affinity data, every time you run such a task. To empty the affinity DB2 files, run the CIUCLR sample job by modifying the following commands:

```
DELETE FROM _schema_.CIU_AFF_EVENTS WHERE APPLID='cicsappl';
DELETE FROM _schema_.CIU_AFF_GRP_DATA WHERE APPLID='cicsappl';
DELETE FROM _schema_.CIU_AFF_CMD_DATA WHERE APPLID='cicsappl';
COMMIT;
```

To empty the affinity VSAM files, start the affinity collector without restoring the data. For more information about the restoring data on startup, see, “Specifying region-specific options: General” on page 108.

Removing duplicate resources from the Collector files

You can use the clean up batch utility to remove duplicate resources which minimizes the volume of 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 names might lead to a large amount of duplicate data being stored for the same command in a particular transaction and program combination.

To avoid collecting unnecessary data, set up a resource compression list table. See, “Creating a resource compression list” on page 77.

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 clean up utility

The CICS IA VSAM clean up utility can remove resources from one particular region or from all regions.

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 113.
- The command types that you specify in the REPTYPE DD statement of the clean up 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 clean up 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 clean up from the dependency VSAM files, edit and run the CIUJCLDL job.

Before you run the CIUJCLDL job, modify the following settings:

JOB accounting parameters

Modify the JOB card statement to meet your site standards.

STEPLIB DD statement

Specify the name of the CICS IA load library in which you have installed the clean up utility program, CIUDEL.

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 clean up 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.

CIUINT2 DD statement

Specify the name of the DB2 dependency data file for the chosen region.

CIUINT3 DD statement

Specify the name of the MQ dependency data file for the chosen region.

CIUINT4 DD statement

Specify the name of the IMS dependency data file for the chosen region.

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.

CIUINT6 DD statement

Specify the resource detail data file for the chosen region.

CIUCNTL DD statement

Specify the name of your CICS IA VSAM control file for the chosen CICS region.

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 clean up 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.

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.

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 clean up utility.

Note: The clean up 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 43.

Generated by CICS Transaction Inter-dependency Utility on

2012/10/20 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 510
--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	YYYYYYY 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	YYYYYYY 3
TSQ_CMDG	ZZZZ 3

Figure 43. Example clean up utility report

Chapter 7. The 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.

For more information about DB2 releases, see DB2 for Linux, UNIX and Windows <http://www-01.ibm.com/software/data/db2/linux-unix-windows/>.

CICS IA also provides the Definition Data Language (DDL) sample to create a UDB on Windows. 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. Use the configuration exec to customize the sample tasks to your site's standards and naming conventions.

Note: If you install the no-charge UDB Express-C, no support is provided by IBM.

This section describes how to configure and load a UDB database.

Configure a new UDB environment

How to create a UDB configuration and access a previous UDB configuration.

Before you configure your UDB environment or the Collector environment, you must first update your settings. For more information about modifying your settings see, “Modify your settings” on page 34.

To create a new UDB configuration, select option 5 from the configuration menu.

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

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

Please select an option from the list below:

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

On the following panel, complete the fields and press Enter to continue.

```

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

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

Please select an option from the list below:

Short or Full Configuration . . . S (S/F)

Please enter UDB configuration name and description:

UDB Configuration . . . . UDBCONF1
Description . . . . . UDB Configuration

```

To configure CICS IA for a UDB database, you need to complete these fields:

SHORT OR FULL CONFIGURATION

Set this option to S (Short), if you want to create a UDB configuration by specifying a short list of required variables. All of the other variables will be set to default values.

Set this option to F (Full), if you want to specify all of the required variables.

REQUIRED

UDB CONFIGURATION

The name of the new UDB configuration.

REQUIRED

DESCRIPTION

A short description of the UDB configuration.

OPTIONAL

Output UDB data sets

OUTPUT DSN FOR UDB SCIUSAMP

The output data set that contains the modified SCIUSAMP members to configure UDB database.

REQUIRED

OUTPUT DSN FOR UDB SCIUDAT1

The output data set that contains the modified SCIUDAT1 members to extract UDB CSV Files.

REQUIRED

OUTPUT DSN FOR UDB SCIUDAT2

The output data set that contains the modified SCIUDAT2 members to extract UDB CSV Files.

REQUIRED

OUTPUT DEVICE TYPE

The device type for the output data sets. Defaults to SYSDA.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed.

HLQ for IA output data sets

This field is displayed on the panel for Short configuration only. It is required and specifies the common values for the IA VSAM FILE QUALIFIER, IA GDG DATA SET QUALIFIER, IA LOG STREAM QUALIFIER and IA CSV FILE QUALIFIER variables. These variables are described later in this topic.

UDB Variables

UDB DATABASE NAME

This variable is the database name. This database name is used when you connect to the database. This value is also required when you connect with the CICS IA plug-in. The name is the database location: for example, CICSIADB.

REQUIRED

UDB TABLE QUALIFIER (SCHEMA)

This variable is the qualifier, which is sometimes referred to as schema for the database objects. This value is also required when you connect with the CICS IA plug-in, for example: CICSIA53.

REQUIRED

UDB CODESET

Specifies the code set to use for data that is entered into this database. The recommended value is 1252.

REQUIRED

UDB TERRITORY

Specifies the territory that is to be used for data that is entered into this database. The recommended value is US.

REQUIRED

UDB DATABASE SERVER DIRECTORY

This variable is the directory on your server where the database is located. The directory that you select must exist on your server, for example: CICSIAUDB.

REQUIRED

CICS IA Variables

IA PRODUCT QUALIFIER

The CICS IA data set qualifier. For a Short configuration, this field is not displayed.

IA VSAM FILE QUALIFIER

The qualifier prefix for the CICS IA VSAM data sets where the collected data is stored. This variable is set to the same value that is used in the CICS configuration.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for "HLQ for IA output data sets".

IA GDG DATA SET QUALIFIER

The qualifier prefix for the GDG data set that is used by the Command Flow option. This variable is set to the same value that is used in the CICS configuration.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for "HLQ for IA output data sets".

IA LOG STREAM QUALIFIER

The stream name qualifier for the CIUMTJNL log stream data sets that are used by the Command Flow option. This variable is set to the same value that is used in the CICS configuration.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for "HLQ for IA output data sets".

IA CSV FILE QUALIFIER

The qualifier prefix for the CICS IA CSV output data sets.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and is set to the value specified for "HLQ for IA output data sets".

IA CSV FILE DATACLASS

The data class that you use for the allocation of the CSV data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA CSV FILE STORAGECLS

The storage class that you use for the allocation of the CSV data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA CSV FILE MNGMNTCLASS

The management class that you use for the allocation of the CSV data sets.

OPTIONAL. For a Short configuration, this field is not displayed.

IA CSV FILE SPACE UNITS

The space units that are used to express the data set allocation size that is required.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to CYLINDERS.

IA CSV FILE PRIMARY QTY

The primary allocation quantity for the CSV files, in cylinders, tracks, KB, or megabytes as selected in the **SPACE UNITS** field.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to 10.

IA CSV FILE SECNDRY QTY

The secondary allocation quantity for the CSV files, in cylinders, tracks, KB, or megabytes as selected in the **SPACE UNITS** field.

REQUIRED for a Full configuration. For a Short configuration, this field is not displayed and defaults to 2.

Configure an existing UDB environment

To select an existing UDB configuration, select option 6 from the configuration menu.

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

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

Please select an option from the list below:

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

It is recommended that you take a copy of the configuration you want to upgrade. Enter the letter c against the region you want to upgrade. This creates a new configuration and the first two letters of the configuration name are replaced with c_.

```
***** CICS Interdependency Analyzer for z/OS - V5R3M0 Row 1 to 6 of 6
Command ==>                                     Scroll ==> CSR

                                CICS IA UDB Customization Function

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

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

Cmd  Configuration  Description
    UDB/EXP          UDB Test
    UDBINC09         UDB V53 INC09
s    c_BINC09        UDB V53 INC09

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

Select the copied region and follow the on-screen instructions.

For a full description of all the configuration values refer to the list in “Configure a new UDB environment” on page 151.

Create a CICS IA UDB database

To create a CICS IA UDB database, follow these steps.

1. Download the required files to your DB2 Windows workstation.
 - You must download the following files to your workstation in ASCII format.
 - SCIUDAT2.UDB(CIUTRANS)
 - SCIUDAT2.UDB(CIUTRCD)
 - SCIUDAT2.UDB(CIUTSCD)
 - SCIUSAMP.UDB(CIUUDBC)
 - SCIUSAMP.UDB(CIUUDBT0)

- SCIOUSAMP.UDB(CIUUDBT1)
- SCIOUSAMP.UDB(CIUUDBT2)
- SCIOUSAMP.UDB(CIUUDBT3)
- SCIOUSAMP.UDB(CIUUDBT4)
- SCIOUSAMP.UDB(CIUUDBT5)
- SCIOUSAMP.UDB(CIUUDBT6)

Note:

- The transfer of these files might create a blank line at the end of each file. If the blank line remains at the end of each file, a warning is issued during the create or load process. It is recommended that you edit and remove the last line if required.
- Depending on the method you use you might need to uncomment the CONNECT statement in the samples below.

2. Create the database:
 - a. Review and edit the sample CIUUDBC.
 - b. Use the DB2 Command Line Processor: `db2 -td; -vf CIUUDBC > CIUUDBC.out`
3. Create the database objects:
 - a. Review and edit the sample CIUUDBT0.
 - b. Use the DB2 Command Line Processor: `db2 -td; -vf CIUUDBT0 > CIUUDBT0.out`
4. Create the DB2 Stored Procedures, functions and triggers:
 - a. Review and edit the sample CIUUDBT1.
 - b. Use the DB2 Command Line Processor: `db2 -td# -vf CIUUDBT1 > CIUUDBT1.out`
5. Load the static DB2 tables:
 - a. Load the CIU_VERSION table.
 - Review and edit the sample CIUUDBT2.
 - Use the DB2 Command Line Processor: `db2 -td; -vf CIUUDBT2 > CIUUDBT2.out`
 - b. Load the rest of the static tables.
 - Create and edit the static.in file:


```
CONNECT TO _udbname_;
SET SCHEMA _udbqual_;
import from CIUTSCD of del insert into _udbqual_.CIU_THREADSafe_CMD;
import from CIUTRANS of del insert into _udbqual_.CIU_TRANSLATORS;
import from CIUTRCD of del insert into _udbqual_.CIU_TRUEEXIT_INFO;
COMMIT;
```
 - Use the DB2 Command Line Processor: `db2 -td; -vf static.in > static.out`

Loading the IVP data into the CICS IA UDB database

CICS IA provides sample data that you can load into your new CICS IA UDB database. You can then connect to the database by using the CICS IA plug-in for CICS Explorer and verify that your UDB database is working correctly.

About this task

Complete the following steps to load the IVP sample data into your CICS IA UDB database.

Note: The *hlq* is the high-level qualifier of the CICS IA SMPE target data sets.

Procedure

1. Review the CIUINDX3 member in the *hlq*.SCIUDAT3 data set.
2. Download the UDB related CSV files from *hlq*.SCIUDAT3 to your workstation in ASCII format.
3. Load the CSV files with IVP sample data into the CICS IA UDB database by using IVP as the collection ID. For more information, see “Update the CICS IA UDB database.”
4. View the IVP sample data in the CICS IA plug-in to verify your UDB database. For more information, see “Viewing the IVP sample data” on page 45.

Update the CICS IA UDB database

When a CICS IA UDB database is created, you can load the data.

To load the data to the database, follow these steps:

1. Run the supplied sample jobs to unload the data from the VSAM files into CSV files. For more information about CSV files, see “Preparing CSV files” on page 158.
2. When the CSV files are created, transfer them in ASCII to your DB2 Windows workstation.

Note:

- The transfer of these files might create a blank line at the end of each file. If the blank line remains at the end of each file, a warning is issued during the create or load process. It is recommended that you edit and remove the last line if required.
 - Depending on the method you use, you might need to uncomment the CONNECT statement in the following samples.
 - The DB2 command IMPORT must be used to load data from CSV files into the tables.
3. Load data from CSV files into the tables.
 - a. Create and edit load.in file:

```
CONNECT TO _udbname_;
SET SCHEMA _udbqual_;
import from affinity.csv of del insert into _udbqual_._CIU_AFF_EVENTS;
import from appl1.csv   of del insert into _udbqual_._CIU_APPLS_DESC;
import from appl2.csv   of del insert into _udbqual_._CIU_APPLS_RESOURCE;
import from cics.csv     of del insert into _udbqual_._CIU_CICS_DATA;
import from cmdflow.csv of del insert into _udbqual_._CIU_CMDFLOW_DATA;
import from cmdindex.csv of del insert into _udbqual_._CIU_CMDFLOW_INDEX;
import from conn.csv     of del insert into _udbqual_._CIU_CONNECTIONS;
import from csect.csv    of del insert into _udbqual_._CIU_CSECT_INFO;
import from csprog.csv   of del insert into _udbqual_._CIU_PROGRAM_INFO;
import from db2.csv      of del insert into _udbqual_._CIU_DB2_DATA;
import from event.csv    of del insert into _udbqual_._CIU_EVENT_DETAIL;
import from exit.csv     of del insert into _udbqual_._CIU_EXIT_INFO;
import from file.csv     of del insert into _udbqual_._CIU_FILE_DETAIL;
import from ims.csv      of del insert into _udbqual_._CIU_IMS_DATA;
import from mq.csv       of del insert into _udbqual_._CIU_MQ_DATA;
```

```

import from natural.csv of del insert into _udbqual_.CIU_NATURAL_DATA;
import from program.csv of del insert into _udbqual_.CIU_PROGRAM_DETAIL;
import from region.csv of del insert into _udbqual_.CIU_REGION_INFO;
import from scandet.csv of del insert into _udbqual_.CIU_SCAN_DETAIL;
import from scansum.csv of del insert into _udbqual_.CIU_SCAN_SUMMARY;
import from tdqueue.csv of del insert into _udbqual_.CIU_TDQUEUE_DETAIL;
import from transid.csv of del insert into _udbqual_.CIU_TRANSID_DETAIL;
import from tsqueue.csv of del insert into _udbqual_.CIU_TSQUEUE_DETAIL;
import from webserv.csv of del insert into _udbqual_.CIU_WEBSERV_DETAIL;
COMMIT;

```

- b. If you are loading the scanner tables, then delete all of the records that are related to the scanned libraries by using the following SQL code:

```

CONNECT TO _udbname_;
SET SCHEMA _udbqual_;
DELETE FROM _udbqual_.CIU_CSECT_INFO WHERE DSNAME = 'DSNAME';
DELETE FROM _udbqual_.CIU_PROGRAM_INFO WHERE DSNAME = 'DSNAME';
DELETE FROM _udbqual_.CIU_SCAN_DETAIL WHERE DSNAME = 'DSNAME';
DELETE FROM _udbqual_.CIU_SCAN_SUMMARY WHERE DSNAME = 'DSNAME';
COMMIT;

```

- c. Use the DB2 command line processor: db2 -td; -vf load.in > load.out
4. When some dependency tables are loaded, reload the CIU_RESOURCE table that is used by the CICS Explorer.
 - Review and edit the sample CIUUDBT3.
 - Use the DB2 command line processor: db2 -td; -vf CIUUDBT3 > CIUUDBT3.out
5. For affinity collection, run the CIUSPAFF stored procedure to load the affinity tables that are used by the CICS Explorer.
 - Review and edit the sample CIUUDBT4.
 - Use the DB2 command line processor: db2 -td; -vf CIUUDBT4 > CIUUDBT4.out
6. To provide the relationships between file resources, the regions that own these resources, and correspondent data sets, run the CIUSPRMF stored procedure.
 - Review and edit the sample CIUUDBT5. The parameter **_collid_** must be the actual value of collection ID.
 - Use the DB2 command line processor: db2 -td; -vf CIUUDBT5 > CIUUDBT5.out
7. To identify Application entry points, run the CIUSPPCP stored procedure to reload the CIU_CICS_CHAINP table by collection ID.
 - Review and edit the sample CIUUDBT6. The parameter **_collid_** must be the actual value of collection ID.
 - Use the DB2 command line processor: db2 -td; -vf CIUUDBT6 > CIUUDBT6.out.

Preparing CSV files

Before you can update a CICS IA UDB database with new data, you need to convert the VSAM files that are 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 tables:

hlq.SCIUSAMP.UDB(CIUUDB)

Prepares QSAM CSV files for all Dependency tables, including the DETAILED tables.

Note:

If you are not collecting DB2 data, you can remove step STEP050 through STEP074.

If you are not collecting MQ data, you can remove step STEP075 through STEP094.

If you are not collecting IMS data, you can remove step STEP095 through STEP112.

hlq.SCIUSAMP.UDB(CIUUDB4)

Prepares QSAM CSV files for the CIU_CMDFLOW_DATA and CIU_CMDFLOW_INDEX tables.

hlq.SCIUSAMP.UDB(CIUUDBAF)

Prepares QSAM CSV files for the CIU_REGION_INFO and CIU_AFF_EVENTS tables.

hlq.SCIUSAMP.UDB(CIUUDBAP)

Prepares QSAM CSV files for the CIU_APPLS_RESOURCE and CIU_APPLS_DESC tables.

hlq.SCIUSAMP.UDB(CIUUDBCS)

Prepares QSAM CSV files for the CIU_CSECT_INFO and CIU_PROGRAM_INFO tables.

hlq.SCIUSAMP.UDB(CIUUDBSD)

Prepares QSAM CSV file for the CIU_SCAN_DETAIL table.

hlq.SCIUSAMP.UDB(CIUUDBSS)

Prepares QSAM CSV file for the CIU_SCAN_SUMMARY table.

When you create CSV files by using CIUUDB, CIUUDB4 and CIUUDBAF jobs you should set the variable UDBPARM before jobs starts. The possible values are:

- NOCONV (default). CSV files are created in readable form and can NOT be used for loading into the CICS IA UDB.
- CONV. CSV files are encoded for loading into the CICS IA UDB.

When you create a new CSV file for the Dependency tables, you can associate the collection by using the COLLECTION_ID column in the tables. You can use the same COLLECTION_ID for each load or select a new one, which is recommended in case of major changes in your application or environment. The CICS IA Explorer plug-in allows 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.

Note: Do not change the length of the collection ID in these samples. It must be 16 characters long.

Table 15. Relationships between the CSV files and the corresponding DB2 tables and the sample jobs that create the CSV files.

Sample Job	CSV file	DB2 table	Comment
CIUUDB	CICS.CSV	CIU_CICS_DATA	
	DB2.CSV	CIU_DB2_DATA	
	MQ.CSV	CIU_MQ_DATA	
	IMS.CSV	CIU_IMS_DATA	

Table 15. Relationships between the CSV files and the corresponding DB2 tables and the sample jobs that create the CSV files. (continued)

Sample Job	CSV file	DB2 table	Comment
	WEBSERV.CSV	CIU_WEBSERV_DETAIL	
	FILE.CSV	CIU_FILE_DETAIL	
	PROGRAM.CSV	CIU_PROGRAM_DETAIL	
	TRANSID.CSV	CIU_TRANSID_DETAIL	
	TDQUEUE.CSV	CIU_TDQUEUE_DETAIL	
	TSQUEUE.CSV	CIU_TSQUEUE_DETAIL	
	EXIT.CSV	CIU_EXIT_INFO	
	EVENT.CSV	CIU_EVENT_DETAIL	
	NATURAL.CSV	CIU_NATURAL_DATA	
	REGION.CSV	CIU_REGION_INFO	
	CONN.CSV	CIU_CONNECTIONS	
CIUUDBAF	REGION.CSV	CIU_REGION_INFO	
	AFFINITY.CSV	CIU_AFF_EVENTS	
CIUUDBAP	APPL1.CSV	CIU_APPLS_DESC	
	APPL2.CSV	CIU_APPLS_RESOURCE	
CIUUDB4	CMDFLOW.CSV	CIU_CMDFLOW_DATA	
	CMDINDEX.CSV	CIU_CMDFLOW_INDEX	
CIUUDBCS	CSPROG.CSV	CIU_PROGRAM_INFO	
	CSECT.CSV	CIU_CSECT_INFO	
CIUUDBSD	SCANDET.CSV	CIU_SCAN_DETAIL	
CIUUDBSS	SCANSUM.CSV	CIU_SCAN_SUMMARY	

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 169

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 113.
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 PURGE MESSAGE, RECEIVE MAP, RECEIVE PARTN, ROUTE, SEND CONTROL, SEND MAP, SEND PAGE, SEND PARTNSET, SEND TEXT

COUNTER
BIF DEEDIT, BIF DIGEST, 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 SESSION, ALLOCATE SYSID, CONNECT PROCESS, CONVERSE CONVID, CONVERSE SESSION, FREE, RECEIVE CONVID, RECEIVE SESSION, SEND CONVID, SEND SESSION

FC DELETE, ENDBR, READ, READNEXT, READPREV, RESETBR, REWRITE, STARTBR, UNLOCK, WRITE

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, DISCARD CORBASERVER, INQUIRE CORBASERVER, PERFORM CORBASERVER, SET CORBASERVER

IDB2 CREATE DB2ENTRY, CREATE DB2TRAN, DISCARD DB2ENTRY, DISCARD DB2TRAN, INQUIRE DB2ENTRY, INQUIRE DB2TRAN, SET DB2ENTRY, SET DB2TRAN

IDJAR
CREATE DJAR, DISCARD DJAR, INQUIRE DJAR, INQUIRE JVMPROFILE, PERFORM DJAR

IFEPI All the FEPI SPI commands listed in Dependency-related CICS FEPI SPI commands detected by the CICS Interdependency Analyzer

IJRNL DISCARD JOURNALNAME, DISCARD JOURNALNUM, INQUIRE JOURNALNAME, INQUIRE JOURNALNUM, SET JOURNALNAME, SET JOURNALNUM

ISFC CREATE FILE, DISCARD FILE, INQUIRE FILE, SET FILE

ISPC CREATE PROGRAM, DISCARD PROGRAM, INQUIRE PROGRAM, SET PROGRAM

ISTD CREATE TDQUEUE, INQUIRE TDQUEUE, SET TDQUEUE

ISTR CREATE TRANSACTION, DISCARD TRANSACTION, INQUIRE TRANSACTION, SET TRANSACTION

ISTS CREATE TSMODEL, DISCARD TSMODEL, INQUIRE TSMODEL, INQUIRE TSPool, INQUIRE TSQNAME, INQUIRE TSQUEUE, SET TSQNAME, SET TSQUEUE

ITCP CREATE TCPIPService, DISCARD TCPIPService, INQUIRE TCPIPService, SET TCPIPService

JRNL WAIT JOURNALNAME, WAIT JOURNALNUM, WRITE JOURNALNAME, WRITE JOURNALNUM

LINK GET64 CONTAINER, LINK, PUT64 CONTAINER

LOAD

LOAD, dynamic COBOL calls

OTHER

ADDRESS CWA, ASSIGN APPLID, CHANGE PASSWORD, CHANGE PHRASE, DUMP TRANSACTION, GETMAIN64, FREEMAIN64, SIGNAL EVENT, SIGNON, SIGNOFF, VERIFY PASSWORD, VERIFY PHRASE

RETURN

RETURN TRANSID

START

ASKTIME, CANCEL, DELAY, FORMATTIME, POST, RETRIEVE, START

TC CHANGE TASK, DEQ, ENQ, SUSPEND, WAIT EVENT

TD DELETEQ TD, READQ TD, WRITEQ TD

TS DELETEQ TS, READQ TS, WRITEQ TS

WEB CONVERTTIME, 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 44 on page 165 shows an example header page and Figure 45 on page 166 an example page of interdependency records.

2011/01/04 - CICS INTERDEPENDENCY ANALYZER (CIU) - Version 320 - Page: 1
 --LIST OF CICS COMMAND GROUPS--

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 44. Example CICS report from the Dependency Reporter—header page

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	DSN8CCD 2011-01-27 09.35.06		Y	QR	
		00000B24 ----	RECEIVE MAP 27	2011-01-27 09.34.34	DSN8CCG 2011-01-27 09.35.29		Y	QR	
		00000D62 ----	LINK PROGRAM 51	2011-01-27 09.34.34	DSN8CC1 2011-01-27 09.36.09		Y	QR	
		00001020 ----	SEND MAP 27	2011-01-27 09.34.34	DSN8CCG 2011-01-27 09.35.24		Y	QR	
		0000106A ----	RETURN TRANSID 27	2011-01-27 09.34.34	D8CS 2011-01-27 09.35.24		Y	QR	

Figure 45. 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'FFFFFFFF') 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 Appendix B, “Correlating Load Module Scanner and Dependency Reporter output to source,” on page 247.
2. An output of X'FFFFFFFF' 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 46 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 46. 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 47 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 47. 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 48 on page 169 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 48. 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 239.
- 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 174.

This rest of this section contains:

- “Requesting a report from the Affinities Reporter” on page 170
- “Output from the Affinities Reporter” on page 170
- “Running the affinity report” on page 174
- “Compressing affinity data” on page 176

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, 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 is shown.

The following report 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	No	No
Total Transactions			:	2			
Total Programs			:	2			

Figure 49. A sample report output by the Affinities Reporter

Notes for the sample report output by the Affinities Reporter:

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'FFFFxxxx') 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 50 shows a sample set of definitions to match the report in “Affinity report” on page 171.

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 200.

```
* 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 D7D6C3C1F14040404040404040404040);
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 D7D6C3C1F24040404040404040404040);
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 50. 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

The threadsafe report can be run from the CICS IA Explorer plug-in or as a batch job using the sample CIUJTSQ2 that can be found in SCIUSAMP.CICS. The report runs the DB2 stored procedure CIUSPTR.

The stored procedure gathers the data that is required to generate the report from the following DB2 tables:

- CIU_CICS_DATA
- CIU_DB2_DATA
- CIU_MQ_DATA
- CIU_IMS_DATA
- CIU_REGION_INFO
- CIU_FILE_DETAIL **1**
- CIU_PROGRAM_DETAIL **1**
- CIU_THREADSafe_CMD **2**
- CIU_SCAN_SUMMARY
- CIU_CONNECTIONS
- **1** To obtain data for the CIU_PROGRAM_DETAIL and CIU_FILE_DETAIL you must select the **DETAILED** collection option when you are configuring your collector options. For more information, see “Specifying region-specific options: API and SPI commands to be monitored” on page 113.
- **2** To populate the CIU_THREADSafe_CMD table you must run the sample job SCIUSAMP.DB2(CIUTSLOD). This job is part of the installation and configuration of CICS IA.

Running the report

To run the threadsafe report edit, review the options and run the CIUJTSQ2 job found in SCIUSAMP.CICS.

The following options are available as input parameters from the CIUOPTS DD card:

COLLECTION_ID

The collection ID that you want to run the report against. An asterisk can be used for wildcard requests, for example, **COLLECTION_ID=***. The default is all the collection IDs.

REGIONNAME

The region from which programs are analyzed. Asterisks can be used for wildcard request, for example, **REGIONNAME=***. The default is all the regions in a PROGRAM table.

PROGRAMNAME

The name of the program or programs that you want to analyze. Asterisks can be used for wild card requests, for example, **PROGRAMNAME=*ab*c**. The default is all the programs in PROGRAM table.

CICSLEVEL

The CICS level for which the threadsafe analysis is considered. The default is the release level of the region for the program.

Note: You can use the **CICSLEVEL** parameter to run a report for a later level of CICS. For example, if you have collected data on a CICS TS V4.1 system you can run the report as if it was CICS TS V5.1 system. This report shows you how many of your commands are now threadsafe at this level.

REPORT

The type of report, for example, SUMMARY or DETAIL. The default is a DETAIL report.

LINESPERPAGE

The maximum number of lines that are written to the report before a page break. The value can be 30 - 99999. The default is 60 lines a page.

Analyzing the report

The report assists in the following aspects of threadsafe analysis.

1. The threadsafe status of the command. The status 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.

2. The report also informs you if you have any of the following types of calls present.

Dynamic call

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

Threadsafe Inhibitor call

An EXEC CICS command that can cause an unsafe affinity between transactions. You must 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 threadsafe from CICS TS V4.2 onwards.

CPMS calls

The calls to the CPMS interface are not threadsafe.

MQ calls

The calls to the CICS WebSphere MQ interface are threadsafe from CICS TS V3.2 onwards.

For more information about the threadsafe collection, see the “Creating a scenario-based collection” topic in the CICS IA plug-in help.

Analyzing a sample threadsafe report

Analyzing the sample output produced by a summary report and a detail report.

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.

```

CICS INTERDEPENDENCY ANALYZER VERSION 5.3.0                                2013/08/09:20.38.38    PAGE    1
THREADSAFE DETAIL LISTING FOR CICS TS

Report options:
  COLLECTION_ID=*                REGIONNAME=*        PROGRAMNAME=*    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 51. Example threadsafe report, header page

```

CICS INTERDEPENDENCY ANALYZER VERSION 5.3.0                                2013/08/09:20.38.38    PAGE    2
THREADSAFE DETAIL LISTING FOR CICS TS 4.2

COLLECTION_ID  APPLID  Program  Linkedit  Execution  Concurrency  APIST  Storage  CICS  LIB Dataset Name  RENT
-----
                                     Date      Key
-----

      CMD  Function      Type      Resource      Offset  Program  Use  Thread-
      Type                                     Length  Count  safe
-----
_collid_  IYDZZ42A TSTPGM00 0001-01-01 USER      QUASIRENT  CICSAPI ACTIVE  0670  CICSIA.D.V51.TEST.LOADLIB  1
      CICS DELETEQ      TSQUEUE      IATSTQ01      6D8      4460      1      Y
      CICS INQUIRE      PROGRAM      TSTPGM01      151E     4460      1      N
      CICS LINK          PROGRAM      TSTPGM02      A8C      4460      1      I
      CICS LOAD          PROGRAM      TSTPGM03      79A      4460      1      Y*
      CICS LOAD          PROGRAM      TSTPGM04      7BE      4460      1      Y*
      CICS SET           JOURNAL      DFHJ03        212      4460      1      N
      CICS WAIT          EVENT        69A           4460      21      N
      CICS WRITE         JOURNAL      03            5A8      4460      83     Y
      CICS WRITEQ        TSQUEUE      IATSTQ1      72C      4460      83     Y
Total CICS calls: 9 Threadsafes: 5 Non-Threadsafes: 3 Indeterminate Threadsafes: 1
Total CPSM calls: 0 Threadsafes: 0 Non-Threadsafes: 0
                  DB2 calls: 0 MQ calls: 0 IMS calls: 0
                  Dynamic Calls: 0 Threadsafes Inhibitor calls: 2

```

Figure 52. Example 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, CPSM, 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.

Note: The asterisk (*) marks the threadsafe inhibitor call commands. These are commands that are potentially threadsafe, however you should investigate the command further because it might prevent a program from being 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.

CPSM calls

The number of **EXEC CPSM** commands invoked by the program.

Threadsafe

The number of **EXEC CPSM** commands invoked by the program that are threadsafe.

Non-Threadsafe

The number of **EXEC CPSM** commands invoked by the program that are not threadsafe.

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.

Threadsafe 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 reports or files:

- 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, 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, 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 described previously, 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.
- To extract scanner information into CSV files for use by a CICS IA UDB database refer to the “Preparing CSV files” on page 158.

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 186.

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:

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.

- 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 has the following content:

- A separate line for the following information about each module in the library:
 - Name
 - Size
 - Language (if determined)
If a load module is created from several source languages, only one language is indicated.
 - Whether it is reentrant.
 - 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
- 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 53 is an example of a summary report produced by the Load Module Scanner.

CICS INTERDEPENDENCY ANALYZER Version 5.3.0							06/24/13	Page	1
LOAD MODULE SCANNER - SUMMARY LISTING OF CICSTLS.CICS1104.LOADLIB									
Module Name	Module Length	Module Language	Re-entrant	Language Version	Possible statements..... Affinities	Dependencies	MVS POSTs	Comment	
DFHPLTPI		CICS TABLE							
DFHPLTSD		CICS TABLE							
EMSTESTA	00001FA8	COBOL II	Y	LE	14	14	0		
EMSTESTB	000014A8	COBOL II	Y	LE	0	0	0		
EMSTESTC	00000A48	COBOL II		Non LE	3	3	0		

CICS INTERDEPENDENCY ANALYZER Version 5.3.0							06/24/13	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 53. 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 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.

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[,DETAILMODS] '
```

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:

Module Name - DB010001 / Load Module Length - 00002E00	Module Entry Point - 00000020				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	
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				
Offset Storage Content (HEX)	EDF DEBUG	Possible Command	Depcy	Affinity	

```

-----
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

```

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

```

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= 'DETAIL1[,DETAILMODS][,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 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.
- The CIUPRINT DD statement; specify the destination of error messages (SYSOUT=*)).

You do not require the IMTMOD DD, dummy, statement for a detailed run.

Chapter 11. Running the CSECT Scanner

The CICS IA 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 the following information:

- 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 the following information:

- 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 table and the CSECT information is added to the CIU_CSECT_INFO table. To extract CSECT Scanner data into the CSV files for use by a UDB, see the sample jobs described in “Preparing CSV files” on page 158.

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 run time. 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 277.

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

To run the CSECT Scanner, you can edit and run the CIUJCLCS job. This job produces a printed report and updates the CIU_PROGRAM_INFO and CIU_CSECT_INFO DB2 tables.

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. CSV specifies that the results of the scan are to be written to the CSV files.
- 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.SCIULOAD, 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 CIUUDBCS job

To run the CSECT Scanner, you can edit and run the CIUJCLSS job. This job produces a printed report and creates CSV files.

Before you run the CIUUDBCS job, change the following as appropriate:

- The JOB accounting parameters
- The PARM keyword of the EXEC statement:
`PARM= ' [CSV] '`
where CSV specifies that the results of the scan are to be written to the CSV files.
- 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.SCIULOAD, where hlq is the high-level data set qualifier assigned 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.
-
-

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 the following information:

- 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 the following information:

- 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.

The following report is an example of a report produced by the CSECT Scanner:

CICS INTERDEPENDENCY ANALYZER Version 3.2.0
CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

10/01/11 Page 1

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	ZAPdate	ZAPdata
CSECT	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData		
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

10/01/11 Page 3

CSECT SCANNER - LISTING OF: CICSTLS.STRESS.LOADLIB

Program	Length	Entry	Alias of	Linker name	Version	Timestamp	AMODE	RMODE	ZAPdate	ZAPdata
CSECT	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData		
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
	Tldate	Tlname	Tlver	T2date	T2name	T2ver	UsrDate	UserData		
CSECT										
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	RS111151198		
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				

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 198
- “Output from the Builder” on page 201

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, 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 200.

- 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 54 on page 200.

```

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 54. 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 55 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(AUXGRP );
CREATE TRANGRP NAME(AUXGRP ) AFFINITY(GLOBAL) AFFLIFE(SYSTEM )
      MATCH(LUNAME) STATE(DORMANT);
  CREATE DTRINGRP TRANGRP(AUXGRP ) TRANID(AUX);
  CREATE DTRINGRP TRANGRP(AUXGRP ) TRANID(CWA1);
```

Figure 55. 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 55 (1) and “Data sets processed report” on page 203.

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 16 through Table 21 on page 203 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 16. 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 17. 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 18. 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 19. 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 20. 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 21. 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.

	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 56. 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 57. 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 58 on page 205, 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 58. 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 = CICSPDN1.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 = CICSPDN1.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 59. Sample error report

Each error is accompanied by a message. For a description of the message, see Appendix D, "Messages and codes," on page 319.

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 214

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.

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.

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 run time 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 re-create 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 TS trace. To switch tracing on and off, use the Global Options Menu.

The Global Options Menu, CIU300, is shown in Collector Global Options Menu panel, CIU300.

CICS TS trace can be obtained using the CICS trace utility program.

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 60. It is usually 00000INT.

CIU150	CICS Interdependency Analyzer for z/OS - V5R1M0		2012/07/31
	Statistics Menu for		01:16:31PM
CICS Sysid : Z518 CICS Applid : IYDZZ518			
CINT state		STOPPED	Collecting Dependencies
Records written last save. : 88		Number of pauses . . . : 0	
Total records on file. . . : 88		Number of saves. . . . : 1	
Date/time of last start. . : 2012/07/30 12:54:17PM			
Date/time of last save . . : 2012/07/30 12:55:00PM			
Date/time of last change . : 2012/07/30 12:54:54PM			
Total time RUNNING : 0000:00:41 (HHHH:MM:SS)			
Total time PAUSED. : (HHHH:MM:SS)			
Table dataspace name . . . : % full			
CICS Sysid: Z518 CICS Applid: IYDZZ518 TermID: TC45			
F1=Help	F2=	F3=End	F4=
F7=	F8=	F9=	F10=
F5=Refresh		F6=	
F11=		F12=	

Figure 60. 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 61 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. 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 61. 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 *mm* are the outstanding reply numbers:

```
/DUMP COMM=(IA DUMP)
/R nn,JOBNAME=CICSTS0A,CONT
/R mm,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 link list” on page 83. 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/http/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.ibmlink.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" 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 22. 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?

Table 22. Data sheet of problem determination information for IBM Support Center (continued)

Information for CICS IA plug-in for CICS Explorer problems
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 the Collector can monitor” 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 239

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 236
- “Non-CICS API commands detected” on page 237

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 236.

CICS API commands

This section lists the presentation commands.

Presentation commands

Authentication commands:

Authentication commands detected:

- EXEC CICS VERIFY PASSWORD
- EXEC CICS CHANGE PASSWORD

- EXEC CICS VERIFY PHRASE
- EXEC CICS VERIFY TOKEN
- EXEC CICS CHANGE PHRASE
- EXEC CICS REQUEST PASSTICKET
- EXEC CICS SIGNOFF
- EXEC CICS SIGNON

Built-in functions commands:

Commands detected:

- EXEC CICS BIF DIGEST RECORD
- EXEC CICS BIF DEEDIT FIELD

BMS commands:

BMS commands detected:

- EXEC CICS PURGE MESSAGE
- 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 PAGE

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.

- RECEIVE PARTITIONSET
- SEND PARTITIONSET
- SEND CONTROL
- SEND TEXT

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 HILIGHT
- 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 BUILD ATTACH

- EXEC CICS EXTRACT ATTACH
- EXEC CICS EXTRACT ATTRIBUTES
- EXEC CICS EXTRACT LOGONMSG
- EXEC CICS EXTRACT TCT
- EXEC CICS ISSUE COPY
- EXEC CICS ISSUE ENDFILE
- EXEC CICS ISSUE ENDOUTPUT
- EXEC CICS ISSUE EODS
- EXEC CICS ISSUE ERASEAUP
- EXEC CICS ISSUE LOAD
- EXEC CICS ISSUE PASS
- EXEC CICS ISSUE PREPARE
- EXEC CICS ISSUE PRINT
- EXEC CICS ISSUE RESET
- EXEC CICS POINT
- EXEC CICS WAIT CONVID (APPC)
- EXEC CICS WAIT SIGNAL

The option specified on the EXEC CICS commands is reported as the resource.

Batch data interchange commands:

Commands detected:

- EXEC CICS ISSUE ADD
- EXEC CICS ISSUE ERASE
- EXEC CICS ISSUE REPLACE
- EXEC CICS ISSUE ABORT
- EXEC CICS ISSUE QUERY
- EXEC CICS ISSUE END
- EXEC CICS ISSUE RECEIVE
- EXEC CICS ISSUE NOTE
- EXEC CICS ISSUE WAIT
- EXEC CICS ISSUE SEND

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

Console support commands:

Console support commands detected:

- EXEC CICS WRITE OPERATOR

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.
- 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 commands:

Commands detected:

- EXEC CICS DUMP TRANSID
- EXEC CICS HANDLE ABEND PROGRAM
- EXEC CICS HANDLE AID
- EXEC CICS HANDLE CONDITION
- EXEC CICS IGNORE CONDITION
- EXEC CICS POP HANDLE
- EXEC CICS PUSH HANDLE

File Control commands:

Commands detected:

- EXEC CICS DELETE FILE()
- EXEC CICS ENDBR 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 ASKTIME
- EXEC CICS CANCEL
- EXEC CICS DELAY
- EXEC CICS FORMATTIME
- EXEC CICS POST
- EXEC CICS RETRIEVE
- EXEC CICS RETURN
- EXEC CICS START INTERVAL
- EXEC CICS START REQID INTERVAL
- EXEC CICS START
- EXEC CICS START ATTACH
- EXEC CICS START BREXIT
- EXEC CICS START TRANSID CHANNEL
- EXEC CICS WAIT EVENT

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 ABEND
 - There is no dependency.
 - The only report is that an ABEND command has been issued.
- EXEC CICS DELETE CHANNEL
- 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.
- EXEC CICS GET CONTAINER
- EXEC CICS GET64 CONTAINER
- EXEC CICS INVOKE APPLICATION
- 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 LOAD
- EXEC CICS MOVE CONTAINER
- EXEC CICS PUT CONTAINER
- EXEC CICS PUT64 CONTAINER
- EXEC CICS QUERY CHANNEL
- EXEC CICS RELEASE
- 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 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.

Security services commands:

Security commands detected:

- EXEC CICS QUERY SECURITY

Syncpoint commands:

Syncpoint commands detected:

- EXEC CICS SYNCPOINT

Task control commands:

CICS commands detected:

- EXEC CICS DEQ
- EXEC CICS ENQ

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.

- EXEC CICS SUSPEND

The program has issued a SUSPEND.

- EXEC CICS CHANGE TASK

The program has issued a CHANGE TASK command.

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.

Trace commands:

Trace commands detected:

- EXEC CICS DUMP TRASACTION
- EXEC CICS ENTER TRACENUM
- EXEC CICS MONITOR

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 CLOSE
- WEB CONVERSE
- CONVERTTIME
- WEB ENDBROWSE
- WEB EXTRACT
- WEB OPEN
- WEB PARSE URL
- WEB READ
- WEB READNEXT
- WEB RECEIVE
- WEB RETRIEVE
- WEB SEND
- 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 INVOKE SERVICE
- EXEC CICS INVOKE WEBSERVICE
- EXEC CICS SOAPFAULT ADD
- EXEC CICS SOAPFAULT CREATE
- EXEC CICS SOAPFAULT DELETE
- EXEC CICS WSACONTEXT BUILD
- EXEC CICS WSACONTEXT GET
- EXEC CICS WSACONTEXT DELETE
- EXEC CICS WSAEPR CREATE

Document services commands:

Commands detected:

- EXEC CICS DOCUMENT CREATE
- EXEC CICS DOCUMENT DELETE
- EXEC CICS DOCUMENT INSERT
- EXEC CICS DOCUMENT RETRIEVE
- EXEC CICS DOCUMENT SET

TCP/IP services commands:

Commands detected:

- EXEC CICS EXTRACT CERTIFICATE
- EXEC CICS EXTRACT TCPIP

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 FREEMAIN
- EXEC CICS FREEMAIN64
- EXEC CICS GETMAIN
If the ADDR option is specified on the command "ADDR" is reported as the resource.
- EXEC CICS GETMAIN64

XML parser commands:

Commands detected:

- EXEC CICS TRANSFORM XFORMTYPE(DATATOXML)
- EXEC CICS TRANSFORM XFORMTYPE(XMLTODATA)

CICS SPI commands

Commands detected:

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

AUTOINSTALL commands:

Commands detected:

- CICS EXEC INQUIRE AUTOINSTALL
- CICS EXEC SET AUTOINSTALL

BRFACILITY commands:

Commands detected:

- CICS EXEC INQUIRE BRFACILITY
- CICS EXEC 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.

Capture commands:

Commands detected:

- CICS EXEC INQUIRE CAPTURESPEC
- CICS EXEC INQUIRE CAPOPTPRED
- CICS EXEC INQUIRE CAPDATAPRED
- CICS EXEC INQUIRE CAPINFOSRCE

CONNECTION commands:

Commands detected:

- CICS EXEC CREATE CONNECTION
- CICS EXEC INQUIRE CONNECTION
- CICS EXEC SET CONNECTION
- CICS EXEC DISCARD CONNECTION

CORBASERVER commands:

Commands detected:

- CICS EXEC CREATE CORBASERVER
- CICS EXEC INQUIRE CORBASERVER
- CICS EXEC SET CORBASERVER
- CICS EXEC PERFORM CORBASERVER
- CICS EXEC DISCARD CORBASERVER
- This dependency is between a program and a local CorbaServer.
- The name of the CorbaServer is reported as the resource.

Other CREATE commands:

Commands detected:

- CICS EXEC CREATE LSRPOOL
- CICS EXEC CREATE MAPSET
- CICS EXEC CREATE PARTITIONSET
- CICS EXEC CREATE SESSIONS
- CICS EXEC CREATE TYPETERM

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

DB2 commands:

Commands detected:

- CICS EXEC CREATE DB2CONN
- CICS EXEC INQUIRE DB2CONN
- CICS EXEC SET DB2CONN
- CICS EXEC DISCARD DB2CONN
- CICS EXEC CREATE DB2ENTRY
- CICS EXEC INQUIRE DB2ENTRY
- CICS EXEC SET DB2ENTRY
- CICS EXEC DISCARD DB2ENTRY
- CICS EXEC CREATE DB2TRAN
- CICS EXEC INQUIRE DB2TRAN
- CICS EXEC SET DB2TRAN
- CICS EXEC DISCARD DB2TRAN
- 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.

- This dependency is between a program and a local DB2TRAN, associated with a DB2ENTRY.
- The name of the DB2TRAN is reported as the resource.

DELETSHIPED commands:

Commands detected:

- CICS EXEC INQUIRE DELETSHIPED
- CICS EXEC SET DELETSHIPED
- CICS EXEC PERFORM DELETSHIPED

DISPATCHER commands:

Commands detected:

- CICS EXEC INQUIRE DISPATCHER
- CICS EXEC SET DISPATCHER

DJAR commands:

Commands detected:

- CICS EXEC CREATE DJAR
- CICS EXEC INQUIRE DJAR
- CICS EXEC PERFORM DJAR
- CICS EXEC 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.

DOCTEMPLATE commands:

Commands detected:

- CICS EXEC CREATE DOCTEMPLATE
- CICS EXEC INQUIRE DOCTEMPLATE
- CICS EXEC SET DOCTEMPLATE
- CICS EXEC DISCARD DOCTEMPLATE

DSNAME commands:

Commands detected:

- CICS EXEC INQUIRE DSNAME
- CICS EXEC SET DSNAME

DUMPDS commands:

Commands detected:

- CICS EXEC INQUIRE DUMPDS
- CICS EXEC SET DUMPDS

ENQMODEL commands:

Commands detected:

- CICS EXEC INQUIRE ENQMODEL
- CICS EXEC SET ENQMODEL
- CICS EXEC DISCARD ENQMODEL

EPADAPTER commands:

Commands detected:

- CICS EXEC INQUIRE EPADAPTER

- CICS EXEC SET EPADAPTER
- CICS EXEC DISCARD EPADAPTER

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
- EXEC CICS INQUIRE EPADAPTERSET
- EXEC CICS SET EPADAPTERSET
- EXEC CICS INQUIRE EPADAPTERINSET

This dependency is between a program and business event processed.

The resource reported is the events.

EXIT commands:

Commands detected:

- CICS EXEC ENABLE EXIT
- CICS EXEC DISABLE EXIT
- CICS EXEC EXTRACT EXIT

FILE commands:

Commands detected:

- CICS EXEC CREATE FILE
- CICS EXEC INQUIRE FILE
- CICS EXEC SET FILE
- CICS EXEC DISCARD FILE
- This dependency is between a program and a local file.
- The file name is reported as the resource.

HOST commands:

Commands detected:

- CICS EXEC INQUIRE HOST
- CICS EXEC SET HOST

Other INQUIRE commands:

Inquire commands detected:

- CICS EXEC INQUIRE ASSOCIATION
- CICS EXEC INQUIRE CFDTPOOL
- CICS EXEC INQUIRE ENQ
- CICS EXEC INQUIRE EXCI
- CICS EXEC INQUIRE EXITPROGRAM
- CICS EXEC INQUIRE IPFACILITY
- CICS EXEC INQUIRE JVMPROFILE
 - This dependency is between a program and a JVM profile.
 - The name of the JVM profile is reported as the resource.

- CICS EXEC INQUIRE MVSTCB
- CICS EXEC INQUIRE OSGIBUNDLE
- CICS EXEC INQUIRE OSGISERVICE
- CICS EXEC INQUIRE REQID
- CICS EXEC INQUIRE RRMS
- CICS EXEC INQUIRE STORAGE
- CICS EXEC INQUIRE STREAMNAME
- CICS EXEC INQUIRE SUBPOOL
- CICS EXEC INQUIRE TSPool
 - This dependency is between a program and a local shared temporary storage pool.
 - The name of the TS pool is reported as the resource.
- CICS EXEC INQUIRE UOWDSNFAIL

IPCONN commands:

Commands detected:

- CICS EXEC CREATE IPCONN
- CICS EXEC INQUIRE IPCONN
- CICS EXEC SET IPCONN
- CICS EXEC 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.

IRC commands:

Commands detected:

- CICS EXEC INQUIRE IRC
- CICS EXEC SET IRC

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

JOURNAL commands:

Commands detected:

- CICS EXEC INQUIRE JOURNALNAME
- CICS EXEC SET JOURNALNAME
- CICS EXEC DISCARD JOURNALNAME
- CICS EXEC CREATE JOURNALMODEL
- CICS EXEC INQUIRE JOURNALMODEL
- CICS EXEC DISCARD JOURNALMODEL
- CICS EXEC INQUIRE JOURNALNUM
- CICS EXEC SET JOURNALNUM
- CICS EXEC DISCARD JOURNALNUM
- This dependency is between a program and a CICS journal.

- The journal name is reported as the resource.

LIBRARY commands:

Commands detected:

- CICS EXEC CREATE LIBRARY
- CICS EXEC INQUIRE LIBRARY
- CICS EXEC SET LIBRARY,
- CICS EXEC DISCARD LIBRARY
- This dependency is between a program and the library resource.
- The library name is reported as the resource.

MODENAME commands:

Commands detected:

- CICS EXEC INQUIRE MODENAME
- CICS EXEC SET MODENAME

MONITOR commands:

Commands detected:

- CICS EXEC INQUIRE MONITOR
- CICS EXEC SET MONITOR

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

NETNAME commands:

Commands detected:

- CICS EXEC INQUIRE NETNAME
- CICS EXEC SET NETNAME

PARTNER commands:

Commands detected:

- CICS EXEC CREATE PARTNER
- CICS EXEC INQUIRE PARTNER
- CICS EXEC DISCARD PARTNER

Other PERFORM commands:

Other PERFORM commands detected:

- CICS EXEC PERFORM DUMP
- CICS EXEC PERFORM ENDAFFINITY
- CICS EXEC PERFORM RESETTIME
- CICS EXEC PERFORM SECURITY
- CICS EXEC PERFORM SHUTDOWN
- CICS EXEC PERFORM SSL REBUILD

PROCESSTYPE commands:

Commands detected:

- CICS EXEC CREATE PROCESSTYPE
- CICS EXEC INQUIRE PROCESSTYPE
- CICS EXEC SET PROCESSTYPE
- CICS EXEC DISCARD PROCESSTYPE

PIPELINE commands:

Commands detected:

- CICS EXEC CREATE PIPELINE
- CICS EXEC INQUIRE PIPELINE
- CICS EXEC SET PIPELINE
- CICS EXEC DISCARD PIPELINE
- CICS EXEC PERFORM PIPELINE
- This dependency is between a program and a PIPELINE.
- The name of the PIPELINE is reported as the resource.

PROFILE commands:

Commands detected:

- CICS EXEC CREATE PROFILE
- CICS EXEC INQUIRE PROFILE
- CICS EXEC DISCARD PROFILE

PROGRAM commands:

Commands detected:

- CICS EXEC CREATE PROGRAM
- CICS EXEC INQUIRE PROGRAM
- CICS EXEC SET PROGRAM
- CICS EXEC DISCARD PROGRAM
- This dependency is between a program and another local program.
- The program name is reported as the resource.

RESYNC command:

STATISTICS commands:

Commands detected:

- CICS EXEC COLLECT STATISTICS
- CICS EXEC EXTRACT STATISTICS
- CICS EXEC INQUIRE STATISTICS
- CICS EXEC SET STATISTICS
- CICS EXEC PERFORM STATISTICS

SYSDUMPCODE commands:

Commands detected:

- CICS EXEC INQUIRE SYSDUMPCODE,
- CICS EXEC SET SYSDUMPCODE

SYSTEM commands:

Commands detected:

- CICS EXEC INQUIRE SYSTEM

- CICS EXEC SET SYSTEM

TASK commands:

Commands detected:

- CICS EXEC INQUIRE TASK
- CICS EXEC SET TASK

TCLASS commands:

Commands detected:

- CICS EXEC INQUIRE TCLASS
- CICS EXEC SET TCLASS

TCPIP commands:

Commands detected:

- CICS EXEC INQUIRE TCPIP
- CICS EXEC SET TCPIP

TCPIPSERVICE commands:

Commands detected:

- CICS EXEC CREATE TCPIPSERVICE
- CICS EXEC INQUIRE TCPIPSERVICE
- CICS EXEC SET TCPIPSERVICE
- CICS EXEC 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.

TDQUEUE commands:

Commands detected:

- CICS EXEC CREATE TDQUEUE
- CICS EXEC INQUIRE TDQUEUE
- CICS EXEC SET TDQUEUE
- CICS EXEC DISCARD 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.

TEMPSTORAGE commands:

Commands detected:

- CICS EXEC INQUIRE TEMPSTORAGE
- CICS EXEC SET TEMPSTORAGE

TERMINAL commands:

Commands detected:

- CICS EXEC ACQUIRE TERMINAL
- CICS EXEC CREATE TERMINAL
- CICS EXEC INQUIRE TERMINAL
- CICS EXEC SET TERMINAL
- CICS EXEC DISCARD TERMINAL

TRACEDEST commands:

Commands detected:

- CICS EXEC INQUIRE TRACEDEST
- CICS EXEC SET TRACEDEST

TRACEFLAG commands:

Commands detected:

- CICS EXEC INQUIRE TRACEFLAG
- CICS EXEC SET TRACEFLAG

TRACETYPE commands:

Commands detected:

- CICS EXEC INQUIRE TRACETYPE
- CICS EXEC SET TRACETYPE

TRANCLASS commands:

Commands detected:

- CICS EXEC CREATE TRANCLASS,
- CICS EXEC DISCARD TRANCLASS
- CICS EXEC INQUIRE TRANCLASS
- CICS EXEC SET TRANCLASS

TRANDUMPCODE commands:

Commands detected:

- CICS EXEC INQUIRE TRANDUMPCODE
- CICS EXEC SET TRANDUMPCODE

TRANSACTION commands:

Commands detected:

- CICS EXEC CREATE TRANSACTION
- CICS EXEC INQUIRE TRANSACTION,
- CICS EXEC SET TRANSACTION,
- CICS EXEC DISCARD TRANSACTION
- This dependency is between a program and a local transaction.
- The transaction name is reported as the resource.

TSMODEL commands:

Commands detected:

- CICS EXEC CREATE TSMODEL
- CICS EXEC INQUIRE TSMODEL
- CICS EXEC 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.

TSQNAME commands:

Commands detected:

- CICS EXEC INQUIRE TSQNAME
- CICS EXEC 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.

TSQUEUE commands:

Commands detected:

- CICS EXEC INQUIRE TSQUEUE
- CICS EXEC 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.

UOW commands:

Commands detected:

- CICS EXEC INQUIRE UOW
- CICS EXEC SET UOW

UOWLINK commands:

Commands detected:

- CICS EXEC INQUIRE UOWLINK
- CICS EXEC SET UOWLINK

URIMAP commands:

Commands detected:

- CICS EXEC CREATE URIMAP
- CICS EXEC INQUIRE URIMAP
- CICS EXEC SET URIMAP
- CICS EXEC DISCARD URIMAP
- This dependency is between a program and a URIMAP.
- The name of the URIMAP is reported as the resource.

VTAM commands:

Commands detected:

- CICS EXEC INQUIRE VTAM
- CICS EXEC SET VTAM

WEB commands:

Commands detected:

- CICS EXEC INQUIRE WEB
- CICS EXEC SET WEB

WEBSERVICE commands:

Commands detected:

- CICS EXEC CREATE WEBSERVICE
- CICS EXEC INQUIRE WEBSERVICE
- CICS EXEC SET WEBSERVICE
- CICS EXEC DISCARD WEBSERVICE
- This dependency is between a program and a WEBSERVICE.
- The name of the WEBSERVICE is reported as the resource.

WORKREQUEST commands:

Commands detected:

- CICS EXEC INQUIRE WORKREQUEST
- CICS EXEC SET WORKREQUEST

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 217.

- 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

The DB2 commands that the collector monitors for potential dependencies.

Commands detected:

- ALTER
- CLOSE
- CREATE
- DELETE
- DESCRIBE
- DROP
- EXECUTE
- EXECUTE IMMEDIATE
- EXPLAIN
- FETCH
- INSERT
- OPEN
- PREPARE
- SELECT
- UPDATE

WebSphere MQ: MQM

The MQ commands that the collector monitors for potential dependencies.

Commands detected:

- MQCLOSE
- MQGET
- MQOPEN
- MQPUT
- MQPUT1
- MQBUFMH
- MQCB
- MQCTL
- MQCRTMH
- MQDLTMH
- MQDLTMP
- MQINQ
- MQINQMP
- MQMHBUF
- MQSETMP
- MQSTAT
- MQSUB
- MQSUBRQ

IMS database: EXEC DLI

The EXEC DLI commands that the collector monitors for potential dependencies.

Commands detected:

- DELETE
- GET NEXT
- GET NEXT IN PARENT
- GET UNIQUE
- INSERT
- REPLACE
- SCHEDULE

IMS database: CBLTDLI, ASMTDLI and PLITDLI calls

The IMS macro calls that the collector monitors for potential dependencies.

Commands detected:

- DELETE
- GET NEXT
- GET NEXT IN PARENT
- GET UNIQUE
- INSERT
- REPLACE
- SCHEDULE

CICSplex System Manager API commands

The CICSplex SM commands that the collector monitors for potential dependencies.

The following CICSplex SM commands are detected.

- ADDRESS
- CANCEL
- CONNECT
- COPY
- CREATE
- DELETE
- DISCARD
- DISCONNECT
- EXPAND
- FEEDBACK
- FETCH
- GET
- GETDEF
- GROUP
- LISTEN
- LOCATE
- MARK
- ORDER
- PERFORM SET
- PERFORM OBJECT
- QUALIFY
- QUERY
- RECEIVE
- REFRESH
- REMOVE

- SET
- SPECIFY FILTER
- SPECIFY VIEW
- TERMINATE
- TRANSLATE
- UNMARK
- UPDATE

Natural commands

The Software AG Natural and ADABAS commands that the collector for potential dependencies.

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 FREEMAIN64
- EXEC CICS GETMAIN64
- EXEC CICS GETMAIN

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 and GETMAIN64 SHARED/FREEMAIN64

- 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.

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

This section describes how to match an API command in a Dependency Reporter report and in a Load Module Scanner detail report with the program source code. It also gives some examples of the procedures described.

Dependency Reporter output

The reported offset of a command is the offset from the start of the load module of the BALR to the CICS stub.

To obtain the offset from the start of the program, subtract the length of the CICS stub from the offset reported. You might also need to subtract the lengths of any additional preceding CSECTs. You can then use the compiler listing to find the command.

Load Module Scanner output

The reported offset of a command is the offset from the start of the load module of the command CICS argument zero.

This offset is a constant and therefore is located in the literal pool for the program. As with the Reporter, subtract the length of the CICS stub and preceding CSECTS to obtain the offset from the start of the program. Then locate the argument zero in the compiler listing. Next, match the argument zero to the command, which involves finding the instruction that referenced the argument zero, using the compiler listing.

Load Module Scanner: Assembler language example

Before the BALR to the CICS stub, the CICS translator generates an LA instruction with the argument zero as source.

For example:

LA 14,=X'020280000807000000000000000000000000000000000000'

To locate the EXEC CICS command, you can match the argument zero in the literal pool with the same argument zero in the LA instruction.

Load Module Scanner: COBOL example

The literal pool in VS COBOL II is part of the Constant Global Table (CGT). Having calculated the offset from the start of the program, subtract the start of the CGT from your calculated offset to obtain the offset within the CGT.

An MVC instruction with the argument zero as the source is in the listing, in this form:

MVC D1(L,R1),D2(R2) DFHEIV0 PGMLIT AT ...

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 62.

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 assembly-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 FDMPT/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 62. 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 assembly 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 269
- “The structure of the Load Module Scanner objects” on page 275
- “The structure of the CSECT Scanner objects” on page 277
- “The structure of the CICS regions objects” on page 279
- “The structure of the Command Flow table objects” on page 297
- “The structure for the Detailed resource objects” on page 281

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 258
- “Dependency views” on page 261

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 or CICSplex SM 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 23. The CIU_CICS_DATA table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APP_NAME	CHAR(64)	Application name
APP_VER1	INTEGER	Major version number
APP_VER2	INTEGER	Minor version number
APP_VER3	INTEGER	Micro version number
APP_OPER	CHAR(64)	The operation associated with the task
TRANSID	CHAR(4)	CICS transaction ID
PROGRAM	CHAR(8)	Currently active CICS program.
FUNCTION	CHAR(24)	EXEC CICS command name, such as READ, WRITE, ^{1 4} or EXEC CPSM command name.

Table 23. The CIU_CICS_DATA table (continued)

Column	Type	Description
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.
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.
USER_DATA1	CHAR(48)	Reserved.
USER_DATA2	CHAR(48)	Reserved.
USER_DATA3	CHAR(48)	Reserved.

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, 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 represent the system identifier (SYSID) of the remote region.
4. For information about Function and Type values, see “Type and Function mapping for monitored commands” on page 301.

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 24. The CIU_DB2_DATA table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.
USER_DATA1	CHAR(48)	Reserved.
USER_DATA2	CHAR(48)	Reserved.

Table 24. The CIU_DB2_DATA table (continued)

Column	Type	Description
USER_DATA3	CHAR(48)	Reserved.

Note:

1. For more information about RESTYPE and FUNCTION, see Table 116 on page 314.
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 that is 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 25. The CIU_IMS_DATA table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.

Table 25. The CIU_IMS_DATA table (continued)

Column	Type	Description
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.
USER_DATA1	CHAR(48)	Reserved.
USER_DATA2	CHAR(48)	Reserved.
USER_DATA3	CHAR(48)	Reserved.

Note:

1. For more information about Function and Type, see Table 117 on page 317.

CIU_MQ_DATA

This table stores information about every unique combination of CICS region, transaction, program, function, and WebSphere MQ resource recorded by the Collector. With this table, you can answer questions such as:

- Which WebSphere MQ resources are used by this transaction?
- If I change this WebSphere MQ resource, which programs and transactions are affected?

Table 26. The CIU_MQ_DATA table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
TRANSID	CHAR(4)	CICS transaction ID.
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}

Table 26. The CIU_MQ_DATA table (continued)

Column	Type	Description
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.
USER_DATA1	CHAR(48)	Reserved.
USER_DATA2	CHAR(48)	Reserved.
USER_DATA3	CHAR(48)	Reserved.

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 about Function and Type, see Table 118 on page 317.
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 27. The CIU_NATURAL_DATA table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name

Table 27. The CIU_NATURAL_DATA table (continued)

Column	Type	Description
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
TRANSID	CHAR(4)	CICS transaction ID.
PROGRAM	CHAR(8)	Current program name.
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.

Table 27. The CIU_NATURAL_DATA table (continued)

Column	Type	Description
USER_DATA1	CHAR(48)	Reserved.
USER_DATA2	CHAR(48)	Reserved.
USER_DATA3	CHAR(48)	Reserved.

Note: For more information about Function and Type, see Table 119 on page 317.

Dependency facilitating tables

Dependency facilitating tables are used by CICS IA to gather information that is required for CICS IA processes.

CIU_APPLS_DESC

The CIU_APPLS_DESC table holds the list of applications and a textual description.

Table 28. The CIU_APPLS_DESC table

Column	Type	Description
APPLIC_CODE	CHAR(8)	Application code.
APPLIC_NAME	CHAR(50)	Application Description.

CIU_APPLS_RESOURCES

The CIU_APPLS_RESOURCES table contains all the transactions and programs that make up an application.

Table 29. 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.

CIU_CICS_CHAINP

The CIU_CICS_CHAINP table allows a join with the following tables:

- CIU_CICS_DATA
- CIU_DB2_DATA
- CIU_MQ_DATA
- CIU_IMS_DATA

The CIU_CICS_CHAINP table shows relationships between programs, so that a query on an initial program can show the resources that it uses, and also any resources that are used by other programs that it calls. The call can be made by using an **EXEC CICS LINK**, an **EXEC CICS XCTL**, or a dynamic **CALL**.

Table 30. The CIU_CICS_CHAINP table

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
FRONT_PROG	CHAR(8)	Calling program.
BACK_PROG	CHAR(8)	Called program.

For example, if PROGA links to PROGB, and PROGB calls PROG3, then this information is added to the table in three records:

```
PROGA : PROGB
PROGA : PROGC
PROGB : PROGC
```

To complete the information that is gathered, further records are added to include immediate dependencies as well as the indirect ones:

```
PROGA : PROGA
PROGB : PROGB
PROGC : PROGC
```

CIU_CICS_CONNP

The CIU_CICS_CONNP table is a temporary table to speed up the building of the CIU_CICS_CHAINP table. It consists of all of the rows from the CIU_CICS_DATA table that record the CALL, LINK, and XCTL commands. It holds the data from columns TRANSID and OBJECT in the CIU_CICS_DATA table. The OBJECT is reduced to an 8 character field to enable indexes to be used efficiently in joins between it and the CIU_CICS_CHAIN table. The table is re-created at each refresh of the CICS IA tables.

Table 31. The CIU_CICS_CONNP table

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
PROGRAM	CHAR(8)	Calling program.
CALLEDPROG	CHAR(8)	Called program.

CIU_CICS_CHAINP_T

The CIU_CICS_CHAINP_T table is a temporary table that is used to build up the entries for the CIU_CICS_CHAINP table, which requires a two stage process. The CIU_CICS_CHAINP_T table has the same layout as the CIU_CICS_CHAINP table.

Threadsafe table

CICS commands that are threadsafe in at least one of the supported CICS TS releases are listed. Use the information in the table when reporting which CICS commands used by a program are threadsafe. Load by running job CIUTSLOD in the SCIUSAMP.DB2 data set.

CIU_THREADSafe_CMD

This table stores resource usage information for all resource types for both web service and program names.

Table 32. The CIU_THREADSafe_CMD table

Field Name	Type	Description
COMMAND	CHAR(24)	The EXEC CICS command name, for example, 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.

Table 32. The CIU_THREADSAFE_CMD table (continued)

Field Name	Type	Description
RESOURCE_TYPE	CHAR(16)	The resource type, for example, 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_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-threadsafe blank not analyzed to allow for table migration
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
CICS_TS42	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V4.2. 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_TS51	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V5.1. 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_TS52	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V5.2. 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_TS53	CHAR(1)	Indicates the threadsafe status of the command for CICS TS V5.3. 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_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 33. 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.

Application dependency views

The application dependency views show gathered data about any CICS, DB2, WebSphere MQ, and IMS application dependency.

V_CIU_CICS_INDS2

This view is a simple DB2 equijoin of CIU_CICS_CHAINP and CIU_CICS_DATA, by using the COLLECTION_ID and BACK_PROG fields in the CIU_CICS_CHAINP table and the COLLECTION_ID and PROGRAM fields in CIU_CICS_DATA table.

You can use the V_CIU_CICS_INDS2 view to display both the direct and indirect CICS commands that are used by the initial program. The following table is used to create views to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 34. View V_CIU_CICS_INDS2

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
FRONT_PROG	CHAR(8)	Calling program.
BACK_PROG	CHAR(8)	Called program.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.

Table 34. View V_CIU_CICS_INDS2 (continued)

Column	Type	Description
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
FUNCTION	CHAR(24)	EXEC CICS command.
TYPE	CHAR(16)	CICS resource type.
OBJECT	CHAR(255)	Resource name.
RMTSYSID	CHAR(4)	Remote SYSID if relevant.
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence.
LAST_RUN	TIMESTAMP	Time of last occurrence.

V_CIU_DB2_INDS2

This view is a simple DB2 equijoin of CIU_CICS_CHAINP and CIU_DB2_DATA, by using the COLLECTION_ID and BACK_PROG fields in the CIU_CICS_CHAINP table and the COLLECTION_ID and PROGRAM fields in CIU_DB2_DATA table.

You can use the V_CIU_DB2_INDS2 view to display both the direct and indirect DB2 commands that are used by the initial program. The following table is used to create views to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 35. View V_CIU_DB2_INDS2

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
FRONT_PROG	CHAR(8)	Calling program.
BACK_PROG	CHAR(8)	Called program.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
FUNCTION	CHAR(24)	EXEC CICS command.
RESTYPE	CHAR(16)	DB2 resource type.
RESNAME	CHAR(255)	Resource name.
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence.
LAST_RUN	TIMESTAMP	Time of last occurrence.

V_CIU_MQ_INDS2

This view is a simple DB2 equijoin of CIU_CICS_CHAINP and CIU_MQ_DATA, by using the COLLECTION_ID and BACK_PROG fields in the CIU_CICS_CHAINP table and the COLLECTION_ID and PROGRAM fields in CIU_MQ_DATA table.

You can use the V_CIU_MQ_INDS2 view to display both the direct and indirect MQ commands that are used by the initial program. The following table is used to create views to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 36. View V_CIU_MQ_INDS2

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
FRONT_PROG	CHAR(8)	Calling program.
BACK_PROG	CHAR(8)	Called program.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
FUNCTION	CHAR(24)	EXEC MQ command.
TYPE	CHAR(16)	MQ resource type.
OBJECT	CHAR(255)	Resource name.
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence.
LAST_RUN	TIMESTAMP	Time of last occurrence.

V_CIU_IMS_INDS2

This view is a simple DB2 equijoin of CIU_CICS_CHAINP and CIU_IMS_DATA, by using the COLLECTION_ID and BACK_PROG fields in the CIU_CICS_CHAINP table and the COLLECTION_ID and PROGRAM fields in CIU_IMS_DATA table.

You can use the V_CIU_IMS_INDS2 view to display both the direct and indirect IMS commands that are used by the initial program. The following table is used to create views to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 37. View V_CIU_IMS_INDS2

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
FRONT_PROG	CHAR(8)	Calling program.
BACK_PROG	CHAR(8)	Called program.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.

Table 37. View V_CIU_IMS_INDS2 (continued)

Column	Type	Description
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
FUNCTION	CHAR(24)	EXEC DLI command.
TYPE	CHAR(16)	IMS resource type.
OBJECT	CHAR(255)	Resource name.
USECOUNT	INTEGER	Number of occurrences.
FIRST_RUN	TIMESTAMP	Time of first occurrence.
LAST_RUN	TIMESTAMP	Time of last occurrence.

Transaction dependency views

The transaction dependency views show data that is gathered about any CICS, DB2, WebSphere MQ, and IMS application dependency.

V_CIU_CICS_TRANSID_DET

This view is a simple DB2 equijoin of the CIU_TRANSID_DETAIL table and the V_CIU_CICS_INDS2 view, by using the INITIAL_PROGRAM field in the CIU_TRANSID_DETAIL table and the FRONT_PROG field in the V_CIU_CICS_INDS2 view.

You can use the V_CIU_CICS_TRANSID_DET view to display both the direct and indirect CICS commands that are used by a transaction and the initial program. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 38. View V_CIU_CICS_TRANSID_DET

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
INITIAL_PROGRAM	CHAR(8)	Initial program.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC CICS command.
TYPE	CHAR(16)	CICS resource type.
OBJECT	CHAR(255)	Resource name.

V_CIU_DB2_TRANSID_DET

This view is a simple DB2 equijoin of the CIU_TRANSID_DETAIL table and the V_CIU_DB2_INDS2 view, by using the INITIAL_PROGRAM field in the CIU_TRANSID_DETAIL table and the FRONT_PROG field in V_CIU_DB2_INDS2 view.

You can use the V_CIU_DB2_TRANSID_DET view to display both the direct and indirect DB2 commands that are used by a transaction and the initial program. The following table is used by the DB2 stored procedures to identify “Application” entry points in the CICS IA Explorer plug-in.

Table 39. View V_CIU_DB2_TRANSID_DET

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
INITIAL_PROGRAM	CHAR(8)	Initial program.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	SQL DLI command.
RESTYPE	CHAR(16)	DB2 resource type.
RESNAME	CHAR(255)	Resource name.

V_CIU_MQ_TRANSID_DET

This view is a simple DB2 equijoin of the CIU_TRANSID_DETAIL table and the V_CIU_MQ_INDS2 field, by using the INITIAL_PROGRAM field in the CIU_TRANSID_DETAIL table and the FRONT_PROG field in the V_CIU_MQ_INDS2 view.

You can use the V_CIU_MQ_TRANSID_DET view to display both the direct and indirect MQ commands that are used by a transaction and the initial program. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 40. View V_CIU_MQ_TRANSID_DET

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.

Table 40. View V_CIU_MQ_TRANSID_DET (continued)

Column	Type	Description
TRANSID	CHAR(4)	CICS Transaction ID.
INITAL_PROGRAM	CHAR(8)	Initial program.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC MQ command.
TYPE	CHAR(16)	MQ resource type.
OBJECT	CHAR(255)	Resource name.

V_CIU_IMS_TRANSID_DET

This view is a simple DB2 equijoin of the CIU_TRANSID_DETAIL table and the V_CIU_IMS_INDS2 view, by using the INITIAL_PROGRAM field in the CIU_TRANSID_DETAIL table and the FRONT_PROG field in the V_CIU_IMS_INDS2 view.

You can the V_CIU_IMS_TRANSID_DET view to display both the direct and indirect IMS commands that are used by a transaction and the initial program. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 41. View V_CIU_IMS_TRANSID_DET

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS Transaction ID.
INITAL_PROGRAM	CHAR(8)	Initial program.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC DLI command.
TYPE	CHAR(16)	IMS resource type.
OBJECT	CHAR(255)	Resource name.

Web service dependency views

The web service dependency views show data that is gathered about any CICS, DB2, WebSphere MQ, and IMS application dependency.

V_CIU_CICS_WEBSERV

This view is a simple DB2 equijoin of the CIU_WEBSERV_DETAIL table and the V_CIU_CICS_INDS2 view, by using the PROGRAM field in the CIU_WEBSERV_DETAIL table and the FRONT_PROG field in the V_CIU_CICS_INDS2 view.

You can use the V_CIU_IMS_TRANSID_DET view to display both the direct and indirect CICS commands that are used by a transaction and the initial

program. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 42. View V_CIU_CICS_WEBSERV

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
NAME	CHAR(32)	The web service name.
PROGRAM	CHAR(8)	The name of the program that implements the web service.
URIMAP	CHAR(8)	The name of the URIMAP.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC CICS command.
TYPE	CHAR(16)	CICS resource type.
OBJECT	CHAR(255)	Resource name.

V_CIU_DB2_WEBSERV

This view is a simple DB2 equijoin of the CIU_WEBSERV_DETAIL table and the V_CIU_DB2_INDS2 view, by using the PROGRAM field in the CIU_WEBSERV_DETAIL table and the FRONT_PROG field in V_CIU_DB2_INDS2 view.

You can use the V_CIU_DB2_WEBSERV view to display both the direct and indirect DB2 commands that are used by a web service. The following table is used by the DB2 stored procedures to identify “Application” entry points in the CICS IA Explorer plug-in.

Table 43. View V_CIU_DB2_WEBSERV

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
NAME	CHAR(32)	The web service name.
PROGRAM	CHAR(8)	The name of the program that implements the web service.
URIMAP	CHAR(8)	The name of the URIMAP.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC SQL command.

Table 43. View V_CIU_DB2_WEBSERV (continued)

Column	Type	Description
RESTYPE	CHAR(16)	DB2 resource type.
RESNAME	CHAR(255)	Resource name.

V_CIU_MQ_WEBSERV

This view is a simple DB2 equijoin of the CIU_WEBSERV_DETAIL table and the V_CIU_MQ_INDS2 view, by using the PROGRAM field in the CIU_WEBSERV_DETAIL table and the FRONT_PROG field in the V_CIU_MQ_INDS2 view.

You can use the V_CIU_MQ_WEBSERV view to display both the direct and indirect MQ commands that are used by a web service. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 44. View V_CIU_MQ_WEBSERV

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
NAME	CHAR(32)	The web service name.
PROGRAM	CHAR(8)	The name of the program that implements the web service.
URIMAP	CHAR(8)	The name of the URIMAP.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC MQ command.
TYPE	CHAR(16)	MQ resource type.
OBJECT	CHAR(255)	Resource name.

V_CIU_IMS_WEBSERV

This view is a simple DB2 equijoin of the CIU_WEBSERV_DETAIL table and the V_CIU_IMS_INDS2 view, by using the PROGRAM field in the CIU_WEBSERV_DETAIL table and the FRONT_PROG field in the V_CIU_IMS_INDS2 view.

You can use the V_CIU_IMS_WEBSERV view to display both the direct and indirect IMS commands that are used by a web service. The following table is used by the DB2 stored procedures to identify “Application” entry points by the CICS IA Explorer plug-in.

Table 45. View V_CIU_IMS_WEBSERV

Column	Type	Description
COLLECTION_ID	CHAR(16)	The collection ID that is assigned when the table is loaded.
APPLID	CHAR(8)	CICS region applid.

Table 45. View V_CIU_IMS_WEBSERV (continued)

Column	Type	Description
PLATFORM	CHAR(64)	Platform name.
APPL_NAME	CHAR(64)	Application name.
APPL_VER1	INTEGER	Major version number.
APPL_VER2	INTEGER	Minor version number.
APPL_VER3	INTEGER	Micro version number.
APPL_OPER	CHAR(64)	The operation that is associated with the task.
NAME	CHAR(32)	The web service name.
PROGRAM	CHAR(8)	The name of the program that implements the web service.
URIMAP	CHAR(8)	The name of the URIMAP.
BACK_PROG	CHAR(8)	Called program.
FUNCTION	CHAR(24)	EXEC DLI command.
TYPE	CHAR(16)	IMS resource type.
OBJECT	CHAR(255)	Resource name.

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 271
- “Affinity views” on page 272

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 46. The CIU_AFF_GRP_DATA table

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 Intertransaction TS Transaction-system.

Table 46. The CIU_AFF_GRP_DATA table (continued)

Column	Type	Description
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.
AFFWORSENE	CHAR(10)	The relation type from which the affinity has worsened from one of the following: <ul style="list-style-type: none"> • BACKGROUND • BAPPL • LINK3270 • LUNAME • USERID.
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 For an explanation of these lifetime values, see "Affinity lifetimes" on page 9.
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.
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.

Table 46. The CIU_AFF_GRP_DATA table (continued)

Column	Type	Description
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 47. 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 48. 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 G4 GETMAIN64 G6 GETMAIN64 unmatched 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 49. 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 49. View V_CIU_AFFINITY (continued)

Column	Type	Description
GROUPTYPE	CHAR(30)	<p>The group of CICS commands used by this transaction, one of the following:</p> <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)	<p>The affinity relation type, one of the following:</p> <ul style="list-style-type: none"> • GLOBAL • BACKGROUND • BAPPL • LINK3270 • LUNAME • USERID.
AFFWORSENE	CHAR(10)	<p>The relation type of the affinity has worsened from one of the following:</p> <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)	<p>The lifetime of the affinity, one of the following:</p> <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 49. 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 50. 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
REENTERABLE	CHAR(1)	Whether this program is reentrant, where Y indicates yes, and blank indicates no.
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 51. 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 51. 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 52. 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_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
CICS_TS41	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V4.1. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe

Table 52. The V_CIU_SCAN_TRDSAFE table (continued)

Column	Type	Description
CICS_TS42	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V4.2. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS51	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V5.1. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS52	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V5.2. Values are: Y command is threadsafe N command is not threadsafe I command is indeterminate threadsafe
CICS_TS53	CHAR(1)	Indicates the threadsafe status if the command is for CICS TS V5.3. 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 279

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 53. 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.

Table 53. The CIU_PROGRAM_INFO table (continued)

Column	Type	Description
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 54. 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.
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 previous tables.

Table 55. 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 PROGLLEN. It shows the link-edit timestamps of the programs that are using CICS resources.

Table 56. 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.
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 57. 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” on page 280

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 58. The CIU_REGION_INFO table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region APPLID.
HOMESYSID	CHAR(4)	CICS region system identifier (SYSID).
CICS_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.
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 whether 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.
DEP_COLL_FIRST_COLLECTED	TIMESTAMP	First time dependency data was collected, in the local time format.
DEP_COLL_LAST_COLLECTED	TIMESTAMP	Last time dependency data was collected, in the local time format.
AFF_COLL_FIRST_COLLECTED	TIMESTAMP	First time affinity data was collected, in the local time format.
AFF_COLL_LAST_COLLECTED	TIMESTAMP	Last time affinity data was collected, in the local time format.

The structure for the Detailed resource objects

This section describes the structure for the Detailed resource objects.

- “File resource table” on page 282
- “Program resource table” on page 284
- “Transaction resource table” on page 286
- “Transient Data queue resource table” on page 289
- “Temporary Storage queue resource table” on page 291
- “Web service resource table” on page 292
- “Threadsafe table” on page 259
- “GLUE and TRUE exit resource table” on page 293

Connections table

Use the information in the CIU_CONNECTIONS table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_CONNECTIONS

This table stores the information about IPCONN-based and CONNECTION-based connections that are defined in the CICS region.

Table 59. The CIU_CONNECTIONS table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region Applid.
HOMESYSID	CHAR(4)	SYSID of local region.
RES_NAME	CHAR(8)	CONNECTION or IPCONN CICS TS resource name
TYPE_INDICATOR	CHAR(4)	Values are the following: CONN for a connection made using the CONNECTION CICS TS resource. IPIC for a connection made using the IPCONN CICS TS resource.
STATUS	CHAR(12)	Connection status. The field name applies to connections made using either the CONNECTION or IPCONN CICS TS resources.
IPCONN_APPLID	CHAR(8)	APPLID parameter of the IP connection.
IPCONN_HOST	CHAR(116)	Name of the remote system or its dotted decimal IP address.
IPCONN_PORT	CHAR(8)	Value is in the range 1 - 65535. This value contains the port number that is to be used for outbound requests on this IPCONN; that is, the number of the port on which the remote system is listening. If the IPCONN is defined with PORT(NO), the value is 'NO'.
IPCONN_TCPIPSERVICE	CHAR(8)	TCPIPSERVICE definition that defines the attributes of the inbound processing for this IPCONN.

Table 59. The CIU_CONNECTIONS table (continued)

Field Name	Type	Description
CONNECTION_NETNAME	CHAR(8)	<p>The name by which the remote system is known to the network, from the NETNAME value that is specified in the CONNECTION definition.</p> <p>ISC connection The NETNAME corresponds to the VTAM APPLID of the remote system.</p> <p>CICS to CICS MRO connection The NETNAME is the name that the remote system uses to log on to DFHIRP, from the APPLID option in the system initialization table (SIT).</p> <p>SPECIFIC EXCI connection The NETNAME is the name of the client program that is passed on the EXCI INITIALIZE_USER command.</p> <p>GENERIC EXCI connection The NETNAME is blank.</p> <p>Indirect connection The NETNAME corresponds to the APPLID, as specified in the SIT APPLID option of the terminal-owning region.</p>
FIRST_RUN	TIMESTAMP	Time of first occurrence, in the local time format.
LAST_RUN	TIMESTAMP	Time of latest observation, in the local time format.

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 60. The CIU_FILE_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
FILE_NAME	CHAR(8)	Name of the file resource.

Table 60. The CIU_FILE_DETAIL table (continued)

Field Name	Type	Description
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.
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).

Table 60. The CIU_FILE_DETAIL table (continued)

Field Name	Type	Description
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.
TABlename	CHAR(8)	The name specified for the coupling facility data table.
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 61. The CIU_PROGRAM_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.

Table 61. The CIU_PROGRAM_DETAIL table (continued)

Field Name	Type	Description
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.
LANGUAGE_DEDUCED	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.

Table 61. The CIU_PROGRAM_DETAIL table (continued)

Field Name	Type	Description
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 62. The CIU_TRANSID_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.

Table 62. The CIU_TRANSID_DETAIL table (continued)

Field Name	Type	Description
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).
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.

Table 62. The CIU_TRANSID_DETAIL table (continued)

Field Name	Type	Description
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
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.

Table 62. The CIU_TRANSID_DETAIL table (continued)

Field Name	Type	Description
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 63. The CIU_TDQUEUE_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.
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.

Table 63. The CIU_TDQUEUE_DETAIL table (continued)

Field Name	Type	Description
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.
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.

Table 63. The CIU_TDQUEUE_DETAIL table (continued)

Field Name	Type	Description
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 64. The CIU_TSQUEUE_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.
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_WEBSESV_DETAIL table to write your own SQL applications to query the tables; these applications must use native SQL queries.

CIU_WEBSESV_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 65. The CIU_WEBSESV_DETAIL table

Field Name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.

Table 65. The CIU_WEBSERV_DETAIL table (continued)

Field Name	Type	Description
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.
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 66. The CIU_EXIT_INFO table

Field Name	Type	Special restrictions
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number

Table 66. The CIU_EXIT_INFO table (continued)

Field Name	Type	Special restrictions
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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 67. 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 68. The V_CIU_TRUEEXIT_INFO table

Field Name	Type	Special restrictions
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
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 69. The CIU_EVENT_DETAIL table

Column	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
ARCHIVE_DATE	TIMESTAMP	Time of archiving.

Table 69. The CIU_EVENT_DETAIL table (continued)

Column	Type	Description
APPLID	CHAR(8)	CICS region applid.
HOMESYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
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.
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.

Table 69. The CIU_EVENT_DETAIL table (continued)

Column	Type	Description
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_CIEVENT		
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 70. The CIU_RESOURCE table

Field name	Type	Description
COLLECTION_ID	CHAR(16)	Collection ID.
APPL_NAME	CHAR(64)	Application name
APPL_VER1	INTEGER	Major version number
APPL_VER2	INTEGER	Minor version number
APPL_VER3	INTEGER	Micro version number
APPL_OPER	CHAR(64)	The operation associated with the task
ARCHIVE_DATE	TIMESTAMP	Time of archiving.
TYPE	CHAR(16)	Resource type
OBJECT	VARCHAR(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 71. CIU_VERSION table

Field name	Type	Description
PROID	CHAR(8)	Product ID“5655-Y22”
DB_APAR_LEVEL	CHAR(7)	APAR level of the IA database
EXP_APAR_LEVEL	CHAR(7)	APAR level of the CICS IA plug-in
EXP_MIN_VER	CHAR(4)	Minimum level of the CICS IA plug-in required
EXP_LATEST_VER	CHAR(4)	Latest version of the CICS IA plug-in required
VER_DESC	CHAR(40)	Description of the latest CICS IA plug-in APAR
VER_CUST_DESC	CHAR(80)	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 that is issued by the transactions. You can write your own SQL applications to query the table; these applications must use native SQL to write the queries.

CIU_CMDFLOW_DATA

This table stores information about every unique detail of the commands that are issued by the transaction.

Table 72. The CIU_CMDFLOW_DATA table

Column	Type	Description
TRACE_ID	CHAR(16)	Unique Command Flow run ID.
CMDFLOW_ID	CHAR(08)	Name of the command flow trace.
APPLID	CHAR(8)	CICS region APPLID.
SYSID	CHAR(4)	SYSID of local region.
PLATFORM	CHAR(64)	Platform name.
APP_NAME	CHAR(64)	Application name.
APP_VER1	INTEGER	Major version number.
APP_VER2	INTEGER	Minor version number.
APP_VER3	INTEGER	Micro version number.

Table 72. The CIU_CMDFLOW_DATA table (continued)

Column	Type	Description
APP_OPER	CHAR(64)	The operation that is associated with the task.
TRANSID	CHAR(4)	CICS transaction ID.
TASKID	CHAR(4)	The transaction task ID, from which the command is started.
DISTRIBUTED_UOW	CHAR(27)	Distributed (Network) unit of work for task (the value of this field is the distributed UOW value that is 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 that starts the command.
OFFSET	CHAR(8)	The offset of the command from the start of the program.
FUNCTION_TYPE	CHAR(8)	The type of command, that is, CICS, CICSplex SM, DB2, DLI, or WebSphere MQ, or the name of the task related user exit (TRUE) if the type of command cannot be determined.
FUNCTION_ID	CHAR(4)	The function ID of the command.
FUNCTION	CHAR(24)	Command name.
TYPE	CHAR(16)	Resource type.
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.
FUNCTION_DESC	CHAR(32)	The description of the command (if known).
PREV_TCBMODE	CHAR(2)	The TCB Mode value that is 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 that is started before a TCB mode switch N Not the last command that is started before a TCB mode switch

Table 72. 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 that is started after a TCB mode switch N Not the last command that is started after a TCB mode switch
TCB_SWITCH_BEFORE_COUNT	INTEGER	Number of TCB mode switches from task start to start of execution of current EXEC CICS command.
TCB_SWITCH_AFTER_COUNT	INTEGER	Number of TCB mode switches from task start to end of execution of current EXEC CICS command.
CICS_VERSION	CHAR(4)	The version of CICS from which the data is recorded.
CMD_TCB_CPUTIME_BEFORE	CHAR(20)	The processor time during which the user task was dispatched by the CICS dispatcher domain, recorded before the current CICS command started. (1)
CMD_TCB_CPUTIME_AFTER	CHAR(20)	The processor time during which the user task was dispatched by the CICS dispatcher domain, recorded after the current CICS command ended. For LINK, the field value is 20C'0'. (1)
CMD_TIME_LOCAL	TIMESTAMP	The time the command was issued, in the local time format.
CMD_RMTSYSID	CHAR(4)	The remote SYSID, if relevant. (2)
CMD_RMTNAME	CHAR(8)	The name by which the resource is known in the remote region.
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.
CMD_OD_FACILTYPE	INTEGER	Originating facility type.
CMD_OD_IPFAMILY	INTEGER	Originating client IP address format.
CMD_OD_CLNTPORT	INTEGER	Originating TCP/IP stack port number.
CMD_PH_COUNT	INTEGER	Previous Hop count.
CMD_OD_TASKID	CHAR(08)	Originating task.
CMD_PH_TASKID	CHAR(08)	Previous Hop task ID.
CMD_OD_STARTTIM	CHAR(21)	Originating task start time.
CMD_PH_STARTTIM	CHAR(21)	Previous Hop task start time.

Table 72. The CIU_CMDFLOW_DATA table (continued)

Column	Type	Description
CMD_STARTTIM	CHAR(21)	This is the time the task on the previous hop started.
CMD_OD_ADPTRID	CHAR(64)	Originating adapter ID.
CMD_OD_ADPTRD1	CHAR(64)	Originating adapter data.
CMD_OD_ADPTRD2	CHAR(64)	Originating adapter data.
CMD_OD_ADPTRD3	CHAR(64)	Originating adapter data.
CMD_OD_APPLID	CHAR(08)	APPLID taken from the origin descriptor associated with this task.
CMD_OD_CLNTIPAD	CHAR(39)	Originating client IP address.
CMD_OD_FACILNM	CHAR(08)	Originating facility name.
CMD_OD_LUNAME	CHAR(08)	Originating VTAM LU name.
CMD_OD_NETID	CHAR(08)	Originating network ID.
CMD_OD_NETWORKI	CHAR(08)	Originating network ID.
CMD_OD_TRANSID	CHAR(04)	Originating transaction ID.
CMD_OD_USERID	CHAR(08)	Originating user ID.
CMD_PH_APPLID	CHAR(08)	Previous Hop APPLID.
CMD_PH_NETWORKI	CHAR(08)	Previous Hop NETWORKID.
CMD_PH_TRANSID	CHAR(04)	Previous Hop transaction ID.

Note: 1. The field value in the database and in the CSV file is a decimal character value. 1 unit (00000000000000000001) is equal to 1/4096000 of a millisecond. The CICS IA plug-in displays the value in milliseconds. If the value is 0 then the CICS IA plug-in displays a blank value.

Note: 2. If the command is shipped or routed to a remote region, these characters represent the system identifier (SYSID) of the remote region.

CIU_CMDFLOW_INDEX

This table contains all instances of the command flows that have run.

Table 73. 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 that is run if specified.
CMD_APPLID2	CHAR(08)	APPLID of the CICS region 2 for this Command Flow that is run if specified.

Table 73. The CIU_CMDFLOW_INDEX table (continued)

Column	Type	Description
CMD_APPLID3	CHAR(08)	APPLID of the CICS region 3 for this Command Flow that is run if specified.
CMD_APPLID4	CHAR(08)	APPLID of the CICS region 4 for this Command Flow that is run if specified.
CMD_APPLID5	CHAR(08)	APPLID of the CICS region 5 for this Command Flow that is run if specified.
CMD_APPLID6	CHAR(08)	APPLID of the CICS region 6 for this Command Flow that is run if specified.
CMD_APPLID7	CHAR(08)	APPLID of the CICS region 7 for this Command Flow that is run if specified.
CMD_APPLID8	CHAR(08)	APPLID of the CICS region 8 for this Command Flow that is run if specified.
CMD_APPLID9	CHAR(08)	APPLID of the CICS region 9 for this Command Flow that is run if specified.
CMD_APPLID10	CHAR(08)	APPLID of the CICS region 10 for this Command Flow that is run if specified.
CMD_APPLID11	CHAR(08)	APPLID of the CICS region 11 for this Command Flow that is run if specified.
CMD_APPLID12	CHAR(08)	APPLID of the CICS region 12 for this Command Flow that is run if specified.
CMD_APPLID13	CHAR(08)	APPLID of the CICS region 13 for this Command Flow that is run if specified.
CMD_APPLID14	CHAR(08)	APPLID of the CICS region 14 for this Command Flow that is run if specified.
CMD_APPLID15	CHAR(08)	APPLID of the CICS region 15 for this Command Flow that is run if specified.
CMD_TRANID1	CHAR(04)	Transaction that is captured by the trace.
CMD_TRANID2	CHAR(04)	Transaction that is captured by the trace.
CMD_TRANID3	CHAR(04)	Transaction that is captured by the trace.
CMD_TRANID4	CHAR(04)	Transaction that is captured by the trace.
CMD_TRANID5	CHAR(04)	Transaction that is captured by the trace.
CMD_COUNT	INTEGER	Number of records that are captured during trace in this region.
CMD_APPLNAME	CHAR(8)	Application name.
CMD_USERID	CHAR(8)	Traced user ID.
CMD_TERMID	CHAR(4)	Traced term ID.
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 116 on page 314.

For a list of the possible combinations of TYPE and FUNCTION values in IMS queries, see Table 117 on page 317.

For a list of the possible combinations of TYPE and FUNCTION values in MQ queries, see Table 118 on page 317.

Table 74. 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 75. 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 76. 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 77. 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 78. 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 79. 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 80. 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 80. 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 INQUIRERSRCE 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 80. 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
	CSDLOCK	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 81. 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 82. 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 83. 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 84. 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 85. 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 86. 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 87. 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 88. 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 89. 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 90. 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 91. 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 92. 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 93. 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 94. Type and Function mapping for monitored commands using the SPI JVMServer CICS resource option flag

Resource type	Function	CICS command name
JVMSEVER	CREATE	CREATE JVMSEVER
	INQUIRE	INQUIRE JVMSEVER
	INQUIRE NEXT	INQUIRE JVMSEVER NEXT
	DISCARD	DISCARD JVMSEVER
	SET	SET JVMSEVER

Table 95. 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 96. 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 97. 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	SYNCPPOINT ROLLBACK
	SYNC	SYNCPPOINT

Table 98. 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 99. 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 100. 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 101. 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 102. 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
CONTAINER	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 103. 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 104. 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 105. Type and Function mapping for monitored commands using the API TD Queues CICS resource option flag

Resource type	Function	CICS command name
TD	DELETEDQ	DELETEDQ TD
	READQ	READQ TD
	WRITEQ	WRITEQ TD

Table 106. 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
TSPOOL	INQUIRE	INQUIRE TSPOOL

Table 107. 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
	START	START INTERVAL
	START	START REQID INTERVAL
	STARTREQ	START REQID

Table 108. 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 109. 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 110. 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
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 111. 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 112. 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 112. 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 113. 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 114. 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 115. 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 116. The possible combinations of TYPE and FUNCTION values in DB2 queries

RESOURCE TYPE	FUNCTION
Dynamic	EXECUTE IMMEDIATE
CURSOR	OPEN
	FETCH
	CLOSE
	ALLOCATE CURSOR

Table 116. 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 116. 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 117. 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 118. 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 119. 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 alphanumeric order.

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 383.
- Collector CINB request queue manager, CIUCINP. See “Collector CINB request queue manager diagnostics” on page 385.
- CICS IA date formatter, CIUCINDT. See “Date formatter diagnostics” on page 385.

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.

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.

CIU1013W 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: CICS IA has not been configured to collect DB2 resource information in this CICS region but the CICS region has a DB2 connection. If you want to collect DB2 resource information, rerun the configuration EXEC for this CICS region and supply the required DB2 variables.

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.

Module: CIUIVPC

Destination

CINT TD queue

**CIU1015W 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: CICS IA has not been configured to collect DB2 resource information in this CICS region but the CICS region has a DB2 connection. If you want to collect DB2 resource information, rerun the configuration EXEC for this CICS region and supply the required DB2 variables.

Module: CIUIVPC

Destination

CINT TD queue.

**CIU1016I 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.

**CIU1017E 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 CEDTA transaction to verify that the required CICS IA resource group is included in the startup group list.
2. Use the CEDTA 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, CIUAWSDA

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, CIUAWSDA, CIUAWSCF

Destination

Terminal end user and CINT TD queue.

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, CIUAWSCF, CIUAWSDA.

Destination

Terminal end user and CINT TD queue.

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/CPSM options are updated

Explanation: The CINT transaction has successfully amended the DB2/MQ/IMS/CPSM 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.

CIU2159E Task collection trigger value must be between 1 and 9999

Explanation: Enter the value between 1 and 9999 or spaces for non-DEFAULTS. This relates to the Dependency collector performance tuning. A value of N indicates that every Nth task will be collected. A value of 1 indicates that all tasks will be collected.

System action: None.

User response: Correct the invalid input.

Module: CIUA260C

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, CIUAWSDA, CIUAWSCF

Destination

Terminal end user and CINT TD queue.

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.

CIU2185I CINT *region-name* Application Collection Options are updated.

Explanation: Application collection options are successfully saved.

System action: Save updated application collection options.

User response: None.

Module: CIUA210C

Destination

Terminal end user.

CIU2186E Maximum *applications-amount* applications can be selected for dependency collection.

Explanation: User tried to save settings with amount of selected applications over the supported *applications-amount*.

System action: Do not save application collection options.

User response: Reduce amount of selected applications to *applications-amount* or lower.

Module: CIUA210C

Destination

Terminal end user.

CIU2186I CINT *region-name* Application Collection Options are updated by user ID.

Explanation: Application collection options for region *region-name* are successfully saved by CICS TS user ID.

System action: Save updated application collection options for CICS TS region *region-name*.

User response: None.

Module: CIUA210C

Destination

CINT TD queue.

CIU2187I Enable collection of Application Data collection-option.

Explanation: Displays chosen **Enable collection of Application Data** collection-option, this message is displayed with message “CIU2186I” on page 330.

System action: None.

User response: None.

Module: CIUA210C

Destination

CINT TD queue.

CIU2188E Wrong option for collection of Application Data.

Explanation: Wrong value in option field for **Enable collection of Application Data**.

System action: Do not save application collection options.

User response: Examine which options are valid by pressing F1. Change the field value to a valid option.

Module: CIUA210C

Destination

Terminal end user.

CIU2188S Memory allocation error at point point-number.

Explanation: Critical dynamic memory error.

System action: The transaction is terminated with the “IUZA” on page 381 abend code.

User response: Contact CICS IA technical support.

Module: CIUA210C

Destination

CINT TD queue.

CIU2189E For DEFAULTS, Application Collection Option must chosen.

Explanation: When saving DEFAULTS application collection options, option field for **Enable collection of Application Data** must not be empty.

System action: Do not save application collection options.

User response: Examine which options are valid by

pressing F1. Change field value to valid option.

Module: CIUA210C

Destination

Terminal end user.

CIU2191I CINT region-name Application Collection Options are set to DEFAULTS

Explanation: Option field for **Enable collection of Application Data** is blank, use Application Collection options from DEFAULTS in this case.

System action: Use DEFAULTS application collection options for current region.

User response: None.

Module: CIUA210C

Destination

Terminal end user.

CIU2192I CINT region-name Appl. Coll. Options are set to DEFAULTS by user ID. (Appl. Coll. is abbreviated because of TDQ limits)

Explanation: **Application Data Collection** options for region-name are set to Defaults by CICS TS user ID.

System action: Use DEFAULTS application collection options for CICS TS region region-name.

User response: None.

Module: CIUA210C

Destination

CINT TD queue.

CIU2193S Error opening CICS IA Applications file.

Explanation: An error occurred while accessing CIUAPPL VSAM file.

System action: The transaction is terminated with the “IUZB” on page 381 abend code.

User response: Check if VSAM IA Application File (CIUAPPL) exists and is properly defined in CICS TS.

Module: CIUA210C

Destination

CINT TD queue.

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

CIU2202S • CIU2208E

*eibr*code. 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, CIUCINBE, CIUCINB1, CIUCINCE, CIUAWSDA, CIUAWSCF

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, CIUAWSDA, CIUAWSCF

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=eibrcode

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 with message IUZ4.

User response: If the control file was created by an earlier release of CICS IA, run any required migration job. If you run a migration job and it does not solve the problem, delete and re-create 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 383. 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 383.

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 383.

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 383. 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 383.

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 dump code of *abend code*, and the Collector is stopped.

User response: If the original abend was issued by

CINT, then there is 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: CIUA170C, CIUA172C, 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 385. 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 385.

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 385.

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 in the message explanation 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 listed in the explanation.

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 383.

Module: CIUCINB2

Destination

CINT TD queue

CIU2229S **UT/TT table update failed**
FUNC=*function code* **REASON**=*reason code*
TABLE=*table number*

Explanation: The transaction CINT received an invalid response when issuing a call to the table manager, CIUTABM, for table *table number*.

System action: The transaction is terminated by abending IUZZ and the Collector is stopped.

User response: The *function code*, *reason code*, and *table number* values can be looked up in "Collector table manager diagnostics" on page 383.

Module: CIUA110C, CIU140C, CIUA180C

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 385.

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=*rcode*

Explanation: Unexpected error detected by CICS dump domain function.

System action: The transaction is terminated by abending IUZ6.

User response: Contact IBM support.

CIU2246E • CIU2253I

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/CPSM options are updated by *userid***

Explanation: The control record for the supplied APPLID is updated with changed DB2, MQ, IMS, or CPSM 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.

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 CINC has abended *abend* in program *program*

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 dump code 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, CIUA173C, 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 Command Flow Collector is already active for a user when an attempt to start the Command Flow collector for the same user is made. If the CIUCNTL control file is shared across the CICS regions, then only one Command Flow collector can be active for a given user at any given time in any of the shared regions.

System action: The second collector start request is terminated because only one active collector is permitted for each user.

User response: Press F7 (Statistics) on the CICS IA Command Flow Options panel to find out which regions the Command Flow collector is already running for the user.

Module: CIUACM60, CIUACM61, CIUAWSCF

Destination

Terminal end user and 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.

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.

CIU2308I CINC Collector is stopped by *uuuu*

Explanation: CINC collector is stopped by the user where:

- *uuuu* - is the user ID.

System action: CINC command flow collection is stopped.

User response: None.

Module: CIUACM71

Destination

CINT TD queue.

CIU2309I CINC Collector has logged *nnnn* records for *uuuu*

Explanation: CINC collector has completed logging for the user, where:

- *nnnn* - is the number of collected records.
- *uuuu* - is the user ID.

System action: CINC collector terminates for the user.

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.

CIU2311I CINC Collector is stopped for *uuuu*.
Limit of tasks: *nnnn* .

Explanation: The specified limit of tasks is reached. CINC collector is stopped for the user, where:

- *uuuu* - is the user ID.
- *nnnn* - is the reached number of tasks.

System action: The CINC command flow collection is stopped.

User response: None.

Module: CIUACM71

Destination

CINT TD queue

**CIU2312I CINC Collector is stopped for *uuuu*.
Limit of records: *nnnn* .**

Explanation: The specified limit of records is reached. CINC collector is stopped for the user, where:

- *uuuu* - is the user ID.
- *nnnn* - is the reached number of records.

System action: The CINC command flow collection is stopped.

User response: None.

Module: CIUACM71

Destination

CINT TD queue

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 *userid*

Explanation: The CINC transaction is already used by user *userid* when an attempt to start another CINC transaction by the same user in the same CICS region is made.

System action: The second CINC transaction is terminated, because only one instance of CINC is permitted for the same user in the same CICS region.

User response: Check where CINC is currently in use. Only one instance of CINC is permitted for the same user in the same CICS region at a given time.

Module: CIUACM10, CIUAWSCF

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 CICS(r) IA Command Flow Statistics panel, CIUA03.

User response: None.

Module: CIUACM70

Destination

Terminal end user.

CIU2321W **APPLIDs *applidID* is not found in control file. Updates are rejected.**

Explanation: The updates are rejected because APPLID *applidID* is not found in the CIUCNTL control file where specified.

System action: The updates are rejected.

User response: Enter correctly the APPLID or delete.

Module: CIUACM20

Destination

Terminal end user.

CIU2322W **Duplicate APPLIDs *applidID* specified. Updates are rejected.**

Explanation: The updates are rejected because duplicate APPLIDs *applidID* 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 and CINT TD queue.

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 instance of the CINC transaction for the user *userid* was not completed successfully, or the CINC transaction is already running in another region or regions for the specified user.

System action: None.

User response: Be careful when changing the Command Flow Options while the CINC transaction is running in a few regions at the same time for the user.

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.

CIU2331W **Privileged Collection not started. In use by another privileged user *userid***

Explanation: The user with privileged authority cannot start the command flow collection because it has been already started by another privileged user *userid*.

System action: The Command Flow Collection start request is terminated.

User response: Wait until the active Command Flow Collector session is terminated. Then try to start the Collector again.

Module: CIUACM61

Destination

CINT TD queue.

CIU2332W **Privileged Collection not started. In use by general users.**

Explanation: The user with privileged authority cannot start the Command Flow Collector because it has been already started by one or more general users.

System action: The Command Flow Collector start request is terminated.

User response: Wait until all of the active Command Flow Collector sessions are terminated. Then try to start the Collector again.

Module: CIUACM61

Destination

CINT TD queue.

CIU2333W **General Collection not started. In use by privileged user *userid***

Explanation: The user with general authority cannot start the Command Flow Collector because it has been already started by a privileged user *userid*.

System action: The Collector start request is terminated.

User response: Wait until the active Command Flow Collector session is terminated. Then try to start the Collector again.

Module: CIUACM61

Destination

CINT TD queue.

CIU2335W **The value for the Authority field is incorrect**

Explanation: The Authority value must be Privileged, General or blank.

System action: None

User response: Enter a correct value in the Authority field.

Module: CIUA410C

Destination

Terminal end user and CINT TD queue.

CIU2336W **Wildcard characters in the UserID field are not allowed for general users**

Explanation: Only users with privileged authority can use wildcard characters in the UserID field.

System action: None.

User response: Enter a correct value in the UserID field or contact your CICS IA Administrator to change your authority from general to privileged.

Module: CIUACM10

Destination

Terminal end user and 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.

CIU2352I Logstream *lname* assigned for CINC user *userid*

Explanation: The log stream *lname* is assigned for CINC user *userid*.

System action: None

User response: None

Module: CIUACM61

Destination

CINT TD queue.

CIU2353E Different Control file names

Explanation: The Collector is not started because of the data set 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 data set 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

Terminal end user and CINT TD queue.

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, CIUAWSDA

Destination

Terminal end user and 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

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 journal 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

Destination

Terminal end user and CINT TD queue

CIU2514I **User *userid* was added.**

Explanation: The user was successfully added.

System action: None.

User response: None.

Module: CIUA410C, CIUA420C

Destination

Terminal end user and CINT TD queue

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

Destination

Terminal end user and CINT TD queue

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

Destination

Terminal end user 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,

Destination

Terminal end user and CINT TD queue.

CIU2518I **User *userid* updated.**

Explanation: The user data is successfully updated.

System action: None.

User response: None.

Module: CIUA440C, CIUAWSCF

Destination

Terminal end user and CINT TD queue.

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

Destination

Terminal end user 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

Destination

Terminal end user 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.

CIU2523I **CICS Interdependency Analyzer for z/OS - *nnnnnn***

Explanation: Version message that indicates a session of CICS IA.

nnnnnn version and release of CICS IA

System action: None

User response: None

Module: CIUA000C, CIUACM10, CIUAWSDA, CIUAWSCF.

Destination

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 IUYYC 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 IUYYE 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 383.

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.

CIU3318I Tasks before stopping:: *nnnn* .

Explanation: CINC Collector run time option, where:

- *nnnn* is the number of tasks before CINC Collector stops.

System action: None.

User response: None.

Module: CIUACM61

Destination

CINT TD queue

CIU3319I Records before stopping: *nnnn* .

Explanation: The CINC Collector run time option, where:

- *nnnn* - is the number of records before the CINC Collector stopped.

System action: None.

User response: None.

Module: CIUACM61

Destination

CINT TD queue

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.

CIU3324I CINB: *count type* records saved

Explanation: Transaction CINB has finished the scan of the one of collector tables in the data space and wrote *count* records to the corresponding data file.

type has one of the following values:

- DEPENDENCY
- AFFINITY
- CICS
- DB2
- MQ
- IMS
- DTP
- MQX
- CICL
- IMSX
- DETAILED
- NATURAL

where CICS, DB2, MQ, IMS, DTP, MQX, CICL, IMSX, DETAILED and NATURAL are types of Dependency tables.

System action: Records given by *count* were saved to the dependency data file.

User response: None.

Module: CIUCINB2

Destination

CINT TD queue.

CIU3325I CINB: DETAILED: *count* records *type* saved

Explanation: Transaction CINB has finished the scan of the Dependency Detailed table in the data space and wrote *count* records to the dependency data file.

type has one of the following values:

- WEBS

- PROG
- FILE
- TRAN
- TDQ
- TSQ
- EXIT
- EVENT

where WEBS, PROG, FILE, TRAN, TDQ, TSQ, EXIT and EVENT are types of Dependency Detailed table.

System action: Records given by count were saved to the dependency data file.

User response: None.

Module: CIUCINB2

Destination

CINT TD queue

CIU3326I **New dependencies total:** *count*

Explanation: The number of new dependencies found since the last start of the dependency collector. If the dependency collector is active and new dependencies were found during collection, this message is issued when the CINB transaction saves data.

System action: None.

User response: None.

Module: CIUCINB1

Destination

CINT TD queue

CIU3327I **New resource details total:** *count*

Explanation: The number of new resource details found since the last start of the dependency collector. If the dependency collector is active and new resource details were found during collection, this message is issued when the CINB transaction saves data.

System action: None.

User response: None.

Module: CIUCINB1

Destination

CINT TD queue

CIU3371I **Transaction CINS is starting**

Explanation: CINS statistics manager task is starting

System action: None.

User response: None.

Module: CIUCINS

Destination

CINT TD queue

CIU3372S **Transaction CINS has abended with abend code in program program.**

Explanation: This message is issued when the CINS HANDLE ABEND exit program is driven to handle a transaction abend that occurred within the CINS transaction. Note that abends can be issued by CINS (codes IUxx) as well as by CICS.

System action: The transaction is terminated with a transaction dump, with a dumpcode of abend code, collector is not stopped.

User response: If the original abend was issued by CINS, 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: CIUCINSE

Destination

CINT TD queue and console.

CIU3373E **Transaction CINS must be initiated by transaction CINT.**

Explanation: The CICS startcode indicates that the CINS transaction has been started by something other than EXEC CICS START with no data.

System action: The transaction is terminated by abend IUYF.

User response: The CINS transaction can only be started by the Collector control transaction CINT.

Module: CIUCINS

Destination

CINT TD queue.

CIU3374I **Transaction CINS has terminated - Collector is stopping.**

Explanation: Transaction CINS received a request to terminate from transaction CINT because the Collector is stopping.

System action: The transaction is terminated.

User response: None.

Module: CIUCINS

Destination

CINT TD queue.

CIU3380I Collector Statistics not available for this period of time.

Explanation: CICS IA cannot retrieve the collection statistics for this period of time. This can occur when the CINT Collector is stopped, or not enough time has passed to fill the selected period with collection statistics.

System action: None.

User response: None.

Module: CIUA150C

Destination

3270.

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. If the response code is INVREQ then check the CICS Master User trace flag, the flag might have been changed to OFF, manually or by another program. Also, use the CETR transaction to check that there is an active trace destination. One of the following trace destinations must be STARTED:

- Internal Trace
- Auxiliary Trace
- GTF trace

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 383.

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:

- SILOSTx: Natural session initialization event is lost.
- STLOSTx: Natural session termination 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.

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.

- EWAOVER: Change the RDCEXIT Natural profile parameter for CIURDCX1 specifying a larger exit work area.
- EWAMISM: None.
- EWAEXHST: None.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU4301S Service *service* failed RCODE=*rcode* REASON=*reason*

Explanation: System *service* TCBTOKEN, DSPSERV or ALESERV fails with return code *rcode* and reason code *reason* while creating of data space.

System action: System action: Natural interface to CICS IA is disabled until the next CICS image restart.

User response: For further details of the problem refer to the *MVS Programming: Assembler Services Reference*.

Module: CIURDCX1

Destination

Natural TD queue and console.

CIU4302S Interface with Natural is disabled.

Explanation: Collector detected the problem with data space creation in CIURDCX1 exit.

System action: Natural interface to CICS IA is disabled until the next CICS image restart.

User response: Refer to the corresponding message CIU4301S in CINT TD queue.

Module: CIUA110C, CIUACM61

Destination

CINT TD queue.

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 list gives possible reasons for the error:

- UNXSID: Unexpected Natural session identifier.
- INVNNL: Invalid Natural Name List.
- INVEWA: Natural SYSRDC exit work area is invalid, possibly the exit work area versions mismatch.
- UNXAOR: Unexpected TOR/AOR error.

System action: The interface is terminated 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 with abend code 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

CINT 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 with abend code 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

CINT 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.

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.

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.

Module: CIUBLD

Destination

Console

CIU5018E **Load of *modname* has failed AC=*abcode* RC=*reason_code***

Explanation: The LOAD macro attempted to load module *modname*. The *abcode* is returned in GPR 1. If the task abended, it is the abend code that would be returned. The *reason_code* is returned in GPR 15 and is the reason code that is associated with the abend.

System action: The job step is terminated with a nonzero return code.

User response: The MVS *abcode* and *reason_code* indicate the cause of the error. Correct the problem and rerun the job.

Module: CIUBLD, CIUU040

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

Job CIUAFFRD has an incorrect value in the PARM parameter in STEP000.

System action: The job is terminated.

User response: Correct the parameter.

Module: CIUU050, CIUU051, CIUU052, CIUU053, CIUU055, CIUU05N, CIUAFFL1, 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, 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, CIUU56, CIUU05N, CIUAFFL1

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

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, 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, CIUU056, CIUUREG

Destination

Console.

CIU6008E **SQL error:** *xxxxxxx*, *yyyyyyyyyy*

Explanation: SQL error occurred, where:

- *xxxxxxx* 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, CIUNTSQ2

Destination

Console.

CIU6010E **File error:** *xxxxxxx*, *yyyyyyyyyy*

Explanation: An I/O error occurred, where:

- *xxxxxxx* 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, 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, 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 CIUJCLCC.

System action: None.

User response: Review the //REGIONS DD card in sample CIUJCLCC.

Module: CIUCFUPD

Destination

Console.

CIU6040E Input record error – add
 region(*aaaaaaaa,ssss,_dumphlq_*)

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*, the SYSID, *ssss*, and, CICS IA Dump HLQ, *_dumphlq_*, 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 correct.

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.

CIU6052I **Unknown record type** *type* **was read from file** *file*. **The record is skipped.**

Explanation: Informational message. Indicates that during affinity VSAM file data formatting a record with unknown type field (byte in position 3) was read. The record was not processed and was not written to output QSAM. This message may occur if the affinity file is corrupted either by user or due to the CICS IA logical error that leads to affinity data inconsistency.

System action: CIUU046 skips current record processing and continues with the next one.

User response: Check if affinity files were not changed by anyone after they were filled with data by the CICS IA collector. Contact the support.

Module: CIUU046

Destination

CIUPRINT DD

CIU6053I **Data records read:** *number*.

Explanation: Informational message for CIUU046 QSAM report.

System action: None.

User response: None.

Module: CIUU046

Destination

CIUPRINT DD

CIU6054I **Data records processed:** *number*.

Explanation: Informational message for CIUU046 QSAM report.

System action: None.

User response: None.

Module: CIUU046

Destination

CIUPRINT DD

CIU6055I **Data records skipped:** *number*.

Explanation: Informational message for CIUU046 QSAM report.

System action: None.

User response: None.

Module: CIUU046

Destination

CIUPRINT DD

CIU6056W *procedure_name* **warning. SQL contention occurred while modifying DB2 tables data.**

Explanation: DB2 SP *procedure_name* received SQLCODE=-911 after SQL INSERT. If such an error occurs more than 3 times then the CIU6058 message is issued and SP terminates.

System action: Reporter program continues, but ends with RC = 4.

User response: Decrease COMMIT COUNT value.

Module: CIUAFFR2

Destination

CIUPRINT DD

CIU6057E *procedure_name* **error. Invalid parameter specified:** *parameter*.

Explanation: The supplied value for one of the input parameters for the DB2 procedure *procedure_name* is invalid. Invalid value is displayed.

System action: Reporter program terminates with RC = 12.

User response: None.

Programmer response: Check if all input parameters have correct values.

Module: CIUAFFR2

Destination

CIUPRINT DD

CIU6058S *procedure_name* **SQL error.**
SQLCODE=*sql_code*
ERRMSG="*db2_error_message*".

Explanation: SQL error occurred while DB2 procedure *procedure_name* runtime.

System action: Reporter program terminates with RC = 12.

User response: None.

Programmer response: follow the instructions described in DB2 SQL reference.

Module: CIUAFFR2

Destination

CIUPRINT DD

CIU6059I **DB2 procedure** *procedure_name* **ended with RC=***return_code*.

Explanation: Informational message issued when DB2 procedure *procedure_name* ended with *rc*<>0. Message issued in pair with 6056/57/58.

System action: None.

User response: None.

Programmer response: None.

Module: CIUAFFR2

Destination

CIUPRINT DD

CIU6064E **CICS IA Dump HLQ must be 1 to 8 characters and non-blank - for DEFAULTS.**

Explanation: An incorrect CICS IA Dump HLQ option is specified in //REGIONS DD card.

System action: None.

User response: Review the entry in the //REGIONS DD card. Ensure that the DUMP HLQ has correct length and is specified for DEFAULTS Region

Module: CIUCFUPD

Destination

Console

CIU6065E **SYSID must be "DFTS" for DEFAULTS.**

Explanation: An incorrect SYSID is specified in //REGIONS DD card for DEFAULTS region.

System action: None.

User response: Change SYSID specified for DEFAULTS region to DFTS.

Module: CIUCFUPD

Destination

Console

CIU6067E **Translation table** *modname* **length is invalid**

Explanation: The length of the translation table in module *modname* is not equal to 256 bytes.

System action: The job step is terminated with a nonzero return code.

User response: Check translation table in module *modname*. Correct the problem and rerun the job.

Module: CIUU040

Destination

Console.

CIU7000I **5655-U86 (C) Copyright IBM Corp. 2001, 2015**

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 occurred in the CICS IA *module* part of the request processing.

System action: The message is sent to the user as the web service response.

User response: Contact the CICS system support team. Look for details in the CICS IA TD Queue.

Module: CIUAWSDA, CIUA172C, CIUA173C

Destination

Web service requester

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: Modify the Collector global options to specify VSAM file sharing. Select option 3 Configure Global Options from the Collector Main Administration Menu, then use the Global Options Menu panel to change the VSAM file sharing option to YES.

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: CIUAWSCF

Destination

Terminal end user.

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: CIUAWSCF

Destination

Terminal end user 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: CIUAWSDA, CIUAWSCF

Destination

CINT TD queue

CIU7052E Region *name* record Control file has an incorrect format.

Explanation: The entry related to <*name*> region in Control file has an incorrect format.

System action: The XML response containing the message is sent to the user.

User response: Run CINT transaction in 3270 mode and then delete or add the region.

Module: CIUAWSDA

Destination

Client, CINT TD queue.

CIU7053E Region *name* record is not found in Control file.

Explanation: The entry related to <*name*> region was not found in Control file.

System action: The XML response containing the message is sent to the user.

User response: Run CINT transaction in 3270 mode and then add the region using the panel user interface.

Module: CIUAWSDA

Destination

Client, CINT TD queue.

CIU7100I CICS IA beta trial period is over.

Explanation: You are attempting to run a version of CICS IA released as part of the Beta program. This program has now finished.

System action: CICS IA terminates.

User response: None.

Module: CIUA000C, CIUACM10

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 **Enter 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 Enter 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 Enter 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.

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

IUZ0

Explanation: An internal error was detected when the CICS IA interface to Natural interacted with Natural SYSRDC.

System action: The current transaction abended.

User response: Contact IBM support.

Module: CIURDCX1

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.

IUZF

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.

IUZF • IUZU

Module: CIUCINT1, CIUCINT2

IUZF

Explanation: An unexpected 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

IUZT

Explanation: An error occurred when CICS IA queried DB2 and the option **Terminate on Abend** was set to Y.

System action: The Collector is stopped. The CINT state is displayed as TERMINATED.

User response: Analyze any previous messages and correct the DB2 querying problem.

Module: CIUCINB0, CIUCINB0, CIUCINB2

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

IUZX

Explanation: The length of the Natural SYSRDC exit work area is too short or invalid.

System action: The CICS IA interface to Natural is stopped.

User response: Contact IBM support.

Module: CIURDCX1

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

IUZZ

Explanation: An unexpected error occurred when calling Collector program CIUTABM to access an element in UT or TT tables in the data space, from transaction CINT.

System action: The Collector is stopped.

User response: Refer to message CIU2229S.

Module: CIUA110C, CIUA140C, CIUA180C

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_GA64	25
AUTM_GM64	26
AUTM_LFA	27
AUTM_LFM	28
AUTM_ICR	29
AUTM_ICM	30
AUTM_SPI	31
AUTM_WAIT	32
AUTM_TT	33
AUTM_UT	34
AUTM_PT	35
AUTM_AT	36
AUTM_LT	37
AUTM_ICP	38
AUTM_BLD_DNT	40
AUTM_BLD_GNT	41
AUTM_BLD_TT	42
AUTM_BLD_MERGED	43
AUTM_APP	99

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 CICS Interdependency Analyzer date formatter, CIUCINDT.

Reason code values

CIUD_NO_DATE	1
CIUD_FORMAT	2

Appendix E. CICS IA space considerations

This appendix contains information on how to calculate the space requirements for the following:

- “Data space allocation” on page 388
- “VSAM data set allocation” on page 391
- “DB2 space allocation” on page 393

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 120 and Table 121 on page 388. These values are used in later calculations.

Table 120. 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.	
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.	

Table 120. Values required for each CICS region (continued)

Required Data	Description	Value
NUM_AFF_PROG	Number of programs that contain EXEC AFF commands.	
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 121. 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:

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. 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 SCIUSAMP.CICS 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, KILOBYTES or MEGABYTES)
- VSAM file primary quantity, in the space units specified
- VSAM file secondary quantity, in the space units specified

Control file: CIUCNTL

Maximum record length is 898 bytes. Number of records is the number of CICS regions in the shared environment and the number of Command Flow user records. Call this number NUM_REGION_V and NUM_CMDF_USERS respectively.

Size = ((898 * NUM_REGION_V) +
 (560 * NUM_CMDF_USERS) + 32)

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 388.

CIUINT1 - CICS

$$\text{Size} = ((109 * \text{NUM_REGION_V}) + (139 * \text{NUM_CICS_PROG} * \text{AVG_EXEC_SHORT} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT2 - DB2

$$\text{Size} = ((122 * \text{NUM_REGION_V}) + (151 * \text{NUM_DB2_PROG} * \text{AVG_DB2} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT3 - MQ

$$\text{Size} = ((163 * \text{NUM_REGION_V}) + (192 * \text{NUM_MQ_PROG} * \text{AVG_MQ} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

CIUINT4 - IMS

$$\text{Size} = ((77 * \text{NUM_REGION_V}) + (106 * \text{NUM_IMS_PROG} * \text{AVG_IMS} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

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

$$\begin{aligned} \text{Size} = & ((128 * \text{NUM_REGION_V}) + \\ & (377 * \text{NUM_CICS_FILE}) + \\ & (280 * \text{NUM_CICS_PROG}) + \\ & (279 * \text{NUM_CICS_TRAN}) + \\ & (325 * \text{NUM_CICS_TDQ}) + \\ & (920 * \text{NUM_CICS_EVENT}) + \\ & (224 * \text{NUM_CICS_TSQ}) + \\ & (1539 * \text{NUM_CICS_WEBS}) + \\ & (176 * \text{NUM_CICS_EXIT})) \text{ bytes} \end{aligned}$$

CIUINT7 - Natural

$$\text{Size} = ((127 * \text{NUM_REGION_V}) + (171 * \text{NUM_NATURAL_PROG} * \text{AVG_NATURAL} * \text{PROG_TRAN_RATIO} * \text{NUM_REGION_V})) \text{ bytes}$$

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)

$$\text{Size} = ((63 * \text{NUM_REGION_V}) + (255 * \text{NUM_AFF_PRG} * (\text{AVG_AFF}/10 * 4) * \text{NUM_REGION_V})) \text{ bytes}$$

CIUAFF3 – Affinity (Key > 32)

$$\text{Size} = ((255 * \text{NUM_REGION_V}) + (255 * \text{NUM_AFF_PRG} * (\text{AVG_AFF}/10) * \text{NUM_REGION_V})) \text{ bytes}$$

Application file: CIUAPPL

Record length is 152 bytes. Number of records is the number of applications that have been defined in region. Call this number APPL_NUMB.

Number of records is the number of application names that have been defined in region:

$$\text{Size} = ((152 * (\text{APPL_NUMB} + 1))) \text{ bytes}$$

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 must 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 following calculation:

$$\begin{aligned} \text{ROWS_PER_PAGE} &= 4089 / \text{ROW_SIZE} \\ \text{FREE_BYTES} &= 4089 / 100 * \text{PERCENT_FREE} \\ \text{FREE_ROWS} &= \text{FREE_BYTES} / \text{ROW_SIZE} \end{aligned}$$

$$\begin{aligned} \text{NUM_OF_4K_PAGES} &= \text{NUM_ROWS} / (\text{ROWS_PER_PAGE} - \text{FREE_ROWS}) \\ \text{PRIQTY} &= 4 * \text{NUM_OF_4K_PAGES} \end{aligned}$$

This calculation has the following 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 index: CIUCICS1 and CIUCICSX” on page 394.

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:

```
NUM_ROWS = (NUM_CICS_PROG * AVG_EXEC_SHORT
             * PROG_TRAN_RATIO * NUM_REGION_D) +
            (NUM_CICS_PROG * AVG_EXEC_LONG
             * PROG_TRAN_RATIO * NUM_REGION_D)
```

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 122 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 122. 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 123 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 123. 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 124 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the DB2 tablespace.

Table 124. 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 125 on page 396 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the MQ tablespace.

Table 125. 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 126 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the IMS tablespace.

Table 126. Worksheet for IMS tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUIMSD	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 127 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the DB2 tablespace.

Table 127. 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 128 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 128. Worksheet for exit resource tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUEXIT	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:

$NUM_ROWS = (NUM_CICS_FILE * NUM_REGION_D)$

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 “DB2 space allocation” on page 393.

Table 129. Worksheet for file detail resource table

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIUFILE	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:

$NUM_ROWS = (NUM_CICS_PROG * NUM_REGION_D)$

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 “DB2 space allocation” on page 393.

Table 130. 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:

$NUM_ROWS = (NUM_CICS_TRAN * NUM_REGION_D)$

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 “DB2 space allocation” on page 393.

Table 131. 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 132 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 132. 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 133 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 133. 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 134 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393.

Table 134. 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 135 on page 401.

- 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 135 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the Affinity tablespace.

Table 135. 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 for the CIU_SCAN_SUMMARY multiplied by the average number of EXEC CICS commands per program.

To obtain these values, run the Load Module Scanner in report mode only; that is, run CIUJCLLS and CIUJCLLD.

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 136 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the Load Module Scanner tablespace.

Table 136. Worksheet for Load Module Scanner tablespace

Tablespace	Row_Size	Percent_Free	PRIQTY	SECQTY
CIULMSD				
Summary	86	15		
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 table space and the DB2 indexes required for the CSECT Module Scanner, estimate the total number of rows.

There are three tables in two table spaces:

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 for the CIU_PROGRAM_INFO, 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 following calculation:

Use the value calculated for NUM_ROWS and the values for ROW_SIZE and PERCENT_FREE in Table 137 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the CSECT Module Scanner table space.

Table 137. Worksheet for CSECT Module Scanner table space

Table space	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 138 to calculate the PRIQTY and SECQTY as described in “DB2 space allocation” on page 393 for the CICS region tablespace.

Table 138. 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 table space 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 table space 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 139 to calculate the PRIQTY and SECQTY as described in Calculation for PRIQTY for table spaces and indexes for the Resource table space.

Table 139. Worksheet for Resource table space using DB2 V7.1

Table space	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 140 on page 404 to calculate the PRIQTY and SECQTY as described in Calculation for PRIQTY for table spaces and indexes for the Resource table space.

Table 140. Worksheet for Resource table space using DB2 V8.1

Table space	Row_Size	Percent_Free	PRIQTY	SECQTY
CIURSRC	273	15		
Indexes				
X01RES01	273	2		
X01RES02	255	2		
X01RES03	8	2		

Appendix F. 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 that is 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 141. 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 141. 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 the following note).	1	
CINS	Drives program CIUCINS for a long running task that handles CICS IA runtime collection statistics.	1	
CINC	Drives program CIUACM10 for the Command Flow feature.	3	

Note:

Authorization can be given by granting the user ID access to the CICS IA batch plan.

On all regions, where you want to collect DB2 data ensure that the user ID, that CICS IA runs under, has GRANT permission to the batch plan created in the sample job SCIUSAMP.CICS(CIUDBNT). This permission enables the background transaction, CINB, to access the SYSIBM.SYSDUMMY1, SYSIBM.SYSPACKSTMT, and SYSIBM.SYSSTMT DB2 tables. In most cases, the CICS default user ID is used. However, in some cases it might be that the PLT user ID is used, if it was started by PLT processing, the user ID of the current CINT transaction, or the Link user ID if the CINT transaction is routed to another CICS region.

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 DB2ENTRY(CICSIAD) GROUP(_groupt_)
  DESCRIPTION(CICS IA DB2 ENTRY)
  ACCOUNTREC(NONE) AUTHTYPE(USER) DROLLBACK(YES)
  PLAN(_db2planc_) 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 35 for further information on defining DB2 resources.

Appendix G. 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 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 you 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 call 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 assist you to manage CIUSPAPP processing and several output parameters that inform about the process completion and errors, if any.

The following table lists all the CIUSPAPP parameters.

Table 142. 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
return-code	OUTPUT	INTEGER	Return code

CIUSPAPP INPUT parameters (calltype, appcode, restype)

The **calltype** parameter specifies queries that can be called by the application. The following table lists all available queries and their description.

Table 143. **calltype** values

calltype value	Description
LIST	Calls for a list of the applications that are 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 three values: PROGRAM, TRANSID, or null.

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 following table.

Table 144. **return-code** values

Return code	Description
0	CIUSPAPP procedure that 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".

Returned result sets

The CIUSPAPP stored procedure returns a number of result sets depending on the input parameters values. A result set is a set of rows, effectively a table, which is associated with a cursor opened in the stored procedure and returned to the caller program. You can access the data that is returned in a result set when you run the SQL ASSOCIATE LOCATORS, followed by the SQL ALLOCATE cursor, and then the SQL FETCH loop as shown in the COBOL example for CIUSPAPP stored procedure.

List applications that are defined to CICS IA

To retrieve a list of user applications that are defined to CICS IA, you must set the input parameters as follows:

calltype = 'LIST'

The other input parameters are ignored and can be set to null values.

This call returns one result set with APPLIC_CODE and APPLIC_NAME columns from CIU_APPLS_DESC table.

List CICS resources that are used by an application

To retrieve a list of CICS resources that are used by one particular application, you must set the input parameters as follows;

calltype = 'CICS'

appcode = application code

The **restype** input parameter is ignored and can be set to null value.

This call returns one result set with TYPE and OBJECT columns from the CIU_CICS_DATA table.

List DB2 resources that are used by an application

To retrieve a list of DB2 resources that are used by one particular application, you must set the input parameters as follows;

calltype = 'DB2'

appcode = application code

The **restype** input parameter is ignored and can be set to null value.

This call returns one result set with **restype** and **resname** columns from the CIU_DB2_DATA table.

List MQ resources that are used by an application

To retrieve a list of MQ resources that are used by one particular application, you must set the input parameters as follows:

calltype = 'MQ'

appcode = application code

The **restype** input parameter is ignored and can be set to null value.

This call returns one result set with TYPE and OBJECT columns from the CIU_MQ_DATA table.

List IMS resources that are used by an application

To retrieve a list of IMS resources that are used by one particular application, you must set the input parameters as follows:

calltype = 'IMS'

appcode = application code

The **restype** input parameter is ignored and can be set to null value.

This call returns one result set with TYPE and OBJECT columns from the CIU_IMS_DATA table

List all transactions and programs that are used by an application

To retrieve a list of transactions and programs that are used by one particular application, you must set the input parameters as follows:

calltype = 'DEFN'
appcode = application code
restype = resource type

The **restype** input parameter must be set to either PROGRAM to list all application programs or to TRANSID to list all application transactions. To list both application programs and transactions, it must be set to null value.

This call returns one result set with APPLIC_CODE, APPLIC_TYPE and APPLIC_RESNAME columns from the CIU_APPLS_RESOURCE table.

List all CICS, DB2, MQ, and IMS resources that are used by an application.

To retrieve a list of all CICS, DB2, MQ, and IMS resources that are used by one particular application, you must set the input parameters as follows:

calltype = 'RES'
appcode = application code

The **restype** input parameter is ignored and can be set to null value.

This call returns four result sets which returned separately for CICS, DB2, MQ, and IMS call types.

CIUSPAPP invocation

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  WV-APPLIC-CODE          PIC X(08).
01  WV-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 call 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 manage CIUSPAFF processing and several output parameters that inform about the process completion and errors, if any.

The following table lists all CIUSPAFF parameters.

Table 145. 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

Table 145. CIUSPAFF parameters (continued)

Parameter name	input/output	Type	Description
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 opens 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 that you must specify for each query type.

When you configure the **qtype** parameter, consider the following points:

- 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 that are 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.

When you have defined the **qtype** value, you can configure the **qarg1** and **qarg2** parameters. The following table provides a list of matching values for the selected query type.

Table 146. 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 147. **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, G4, G6, 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 following table.

Table 148. **rc** values

Return code	Description
0	CIUSPAFF procedure that completed successfully.
4	CIUSPAFF procedure that completed successfully, but one or more SQL warning conditions were received during SQL statements execution.
8	CIUSPAFF procedure failed because of a critical error caused by incorrect parameter values.
12	CIUSPAFF procedure failed because of 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 following table.

Table 149. **sqlcode** values

Return code	sqlcode value
0	0

Table 149. **sqlcode** values (continued)

Return code	sqlcode value
4	Shows SQLCODE for the last statement that caused SQL warning condition.
8	0
12	Shows SQLCODE of the failed SQL statement.

Returned result sets

The CIUSPAFF stored procedure returns result sets. The number and structure of the result sets is dependent on the input parameters values. A result set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop as shown in the COBOL example for CIUSPAFF stored procedure. The following subsections describe result sets that are returned by the CIUSPAFF procedure.

Build affinity groups tables

To build affinity groups tables, CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA, you must set the input parameters as follows:

qtype = 'BLD'
qarg1 = applid

If you want to build affinity groups for all CICS TS regions set **qarg1** to %%%%%%%%%.

The **qarg2** input parameter is ignored and can be set to null value.

This call returns no result sets, it processes data from the CIU_AFF_EVENTS table and creates new, or updates the existing affinity groups data in the CIU_AFF_GRP_DATA and CIU_AFF_CMD_DATA tables.

List affinity groups of the specified type for the specified CICS TS region

To receive a list of existing affinity groups of the specified type for the specified CICS TS region, you must set the input parameters as follows:

qtype = 'RGN'
qarg1 = applid
qarg2 = affinity group ID mask

If you want to build affinity groups for all CICS TS regions set **qarg1** to %%%%%%%%%.

This call returns one result set with APPLID, TRANGROUP, AFFTYPE, GROUPTYPE, AFFINITY, AFFWORSENER, LIFETIME, LIFEWORSENER, RECOVERY, RESOURCE, RESLENGTH, and TYPE columns from the CIU_AFF_GRP_DATA table.

List affinity groups of the specified type that contain the specified program

To receive a list of existing affinity groups of the specified type, and which contains a specified program, you must set the input parameters as follows:

qtype = 'PGM'
qarg1 = program name
qarg2 = affinity group ID mask

This call returns one result set with APPLID, TRANGROUP, AFFTYPE, GROUPTYPE, AFFINITY, AFFWORSENER, LIFETIME, LIFEWORSENER, RECOVERY, RESOURCE, and TYPE columns from the CIU_AFF_GRP_DATA.

List affinity groups of the specified type that contains the specified transaction

To receive a list of existing affinity groups of the specified type, and which contains the specified transaction ID, you must set the input parameters as follows:

```
qtype = 'GRP'
qarg1 = transaction ID
qarg2 = affinity group ID mask
```

If you want to build affinity groups for all CICS TS regions, set **qarg1** to %%%%%%%%%%.

This call returns one result set with APPLID, TRANGROUP, AFFTYPE, GROUPTYPE, AFFINITY, AFFWORSENER, LIFETIME, LIFEWORSENER, RECOVERY, RESOURCE, and TYPE columns from the CIU_AFF_GRP_DATA.

List commands for the specified affinity group in the specified region

To receive a list of commands for the existing affinity group, you must set the input parameters as follows:

```
qtype = 'TRN'
qarg1 = applid
qarg2 = affinity group ID
```

If you want to build affinity groups for all CICS TS regions set **qarg1** to %%%%%%%%%%.

This call returns one result set with APPLID, TRANSID, PROGRAM, OFFSET, COMMAND, RESTYPE, AFFGROUP, TERMINAL, BTS, LINK3270, and USAGE columns from the CIU_AFF_CMD_DATA.

CIUSPAFF invocation

Below you can see an example of CISPAFF 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
                    TO 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
.

```

CIUSPDPG Stored Procedure

By using the CIUSPDPG stored procedure you can list and then delete records from CICS IA DB2 tables, which might contain information about old version of programs.

The CIUSPDPG stored procedure works with the following DB2 tables:

- CIU_CICS_DATA
- CIU_DB2_DATA
- CIU_IMS_DATA
- CIU_MQ_DATA
- CIU_NATURAL_DATA

What is the CIUSPDPG Stored Procedure?

Use the CIUSPDPG stored procedure to list CICS TS applications for which you have collected data. Then, you can delete the redundant records that are identified by the CIUSPDPG stored procedure when using the list option.

Syntax

You can invoke the CIUSPDPG procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPDPG (calltype, collid, applid, program-name, table-name, count,  
               return-code, error-message);
```

Procedure parameters

There are several input parameters that can help you to manage the CIUSPDPG processing. The output parameters inform you about the process completion and if there are any errors.

The following table lists all of the CIUSPDPG parameters.

Table 150. CIUSPDPG parameters

Parameter name	input/output	Type	Description
calltype	INPUT	CHAR(1)	Type of call (L or D)
collid	INPUT	VARCHAR(16)	Collection ID
applid	INPUT	VARCHAR(8)	Application ID
program-name	INPUT	VARCHAR(8)	Program Name
table-name	INPUT	VARCHAR(20)	Table Name
count	OUTPUT	INTEGER	Count of records to be deleted
return-code	OUTPUT	INTEGER	Return code
error-message	OUTPUT	CHAR(300)	Error message text

CIUSPDPG INPUT parameters (**calltype**, **collid**, **applid**, **program-name**, **table-name**)

The following table describes the parameters and their matching values.

Table 151. Input parameters

Parameter	Description and values
calltype	Mandatory parameter Defines the type of CIUSPDPG usage Values: L D
collid	Mandatory parameter if calltype is L The collection ID parameter specifies the CICS IA collection ID under which data was collected. Wildcard masks are not permitted.
applid	Mandatory parameter if calltype is L. The applid parameter specifies the CICS region in which the program to be analyzed was running. Wildcard masks are not permitted.
program-name	Mandatory parameter if calltype is L. The program-name parameter specifies whether the program Wildcards % and _ can be used.

Table 151. Input parameters (continued)

Parameter	Description and values
table-name	<p>Mandatory parameter if calltype is L</p> <p>The table-name parameter defines the CICS IA DB2 table to be analyzed.</p> <p>Wildcard masks are not permitted.</p> <p>Values: CIU_CICS_DATA CIU_DB2_DATA CIU_IMS_DATA CIU_MQ_DATA CIU_NATURAL_DATA</p>

CIUSPDGP OUTPUT parameters (count, return-code, error-message)

The **count** parameter contains the number of record to be deleted.

The **return-code** parameter contains the value of the CIUSPDGP return code. Possible **return-code** values are listed in the following table.

Table 152. return-code values

Return code	Description
0	CIUSPDGP procedure completed successfully.
4	CIUSPDGP procedure completed successfully, but one or more SQL warning conditions were received during SQL statements execution.
8	CIUSPDGP procedure failed because of a critical error caused by incorrect parameter values.
12	CIUSPDGP procedure failed because of a disaster error caused by SQL exception conditions during SQL statement execution.

The **error-message** parameter contains message text that describes the error or warning:

- For **return-code** = 4, it returns the last SQL warning that occurred during the CIUSPDGP run time.
- For **return-code** = 8, it returns the invalid parameter value that caused the error. The incorrect parameter can also be found in SQLSTATE (SQLCA):
 - 99999: Invalid calltype specified
 - 99997: Invalid table_name value specified
- For **return-code** = 12, it returns the SQL error message for the failed SQL statement.

CIUSPTSR Stored Procedure

With the help of this CICS IA External Interface, you can produce threadsafe reports directly from your application.

What is the CIUSPTSR Stored Procedure?

CIUSPTSR is a DB2 Stored Procedure that gathers threadsafe information for the specified programs. It queries the CICS IA interdependency database, gets all necessary data, and returns either a summary or detailed threadsafe information for the specified programs in a result set. It can be called from a user application with a SQL CALL statement.

Syntax

To call the CIUSPTSR procedure with a SQL CALL statement, use this syntax:

```
EXEC SQL  
CALL CIUSPTSR (ctype, collid, applid, qarg, cicslevel, rc,  
               sqlcode, errmsg);
```

Procedure parameters

There are several input parameters that manage the CIUSPTSR processing and several output parameters that inform about the process completion and errors, if any.

The following table lists all CIUSPTSR parameters.

Table 153. CIUSPTSR parameters

Parameter name	input/output	Type	Description
ctype	INPUT	CHAR(4)	Call type
collid	INPUT	CHAR(16)	CICS IA collection identifier (COLLECTION_ID)
applid	INPUT	CHAR(8)	CICS TS region APPLID
qarg	INPUT	VARCHAR(8)	Query argument
cicslevel	INPUT	CHAR(4)	Version of the CICS TS
rc	OUTPUT	INTEGER	Return code
sqlcode	OUTPUT	INTEGER	SQLCODE
errmsg	OUTPUT	VARCHAR(300)	Error message text

CIUSPTSR INPUT parameters (ctype, collid, applid, qarg, cicslevel)

Table 2 describes the parameters and their matching values.

Table 154. Input parameters

Parameter	Description and values
ctype	<p>Mandatory parameter.</p> <p>The ctype parameter defines the type of the returned program threadsafe information, either summary or detailed, and the method which is used to gather this information, that is by specified program or by transaction.</p> <p>Values:</p> <p>PGMS For a summary threadsafe information for the specified program.</p> <p>TRNS For a summary threadsafe information for the programs that are started by the specified transaction.</p> <p>PGMD For a detailed threadsafe information , a list of called commands, for the specified program.</p>

Table 154. Input parameters (continued)

Parameter	Description and values
collid	<p>Mandatory parameter.</p> <p>The collection ID parameter specifies the CICS IA collection ID under which data was collected.</p> <p>A wildcard mask, "%", is valid only for call types PGMS and TRNS.</p>
applid	<p>Mandatory parameter.</p> <p>The applid parameter specifies the CICS region in which the program or transaction to be reported on was running.</p> <p>A wildcard mask, "%", is valid only for call types PGMS and TRNS.</p>
qarg	<p>Mandatory parameter.</p> <p>The query argument must specify either the program name or the transaction, depending on the specified ctype parameter.</p> <p>Wildcard masks are not permissible.</p>
cicslevel	<p>Optional parameter.</p> <p>The cicslevel parameter defines the version of CICS TS to be used to determine the thread-safe status of the commands that are issued by the specified program.</p> <p>Values:</p> <p>null or blank</p> <p>Produce a thread-safe report that is based on the thread-safe status of the commands of the CICS TS version of the region on which the specified program was run and collected.</p> <p>3.1</p> <p>Produce a thread-safe report that is based on the thread-safe status of the CICS TS 3.1 commands.</p> <p>4.1</p> <p>Produce a thread-safe report that is based on the thread-safe status of the CICS TS 4.1 commands.</p> <p>4.2</p> <p>Produce a thread-safe report that is based on the thread-safe status of the CICS TS 4.2 commands.</p> <p>5.1</p> <p>Produce a thread-safe report that is based on the thread-safe status of the CICS TS 5.1 commands.</p> <p>5.2</p> <p>Produce a thread-safe report that is based on the thread-safe status of the CICS TS 5.2 commands.</p>

CIUSPTSR OUTPUT parameters (rc, sqlcode, errmsg)

The **rc** parameter contains value of the CIUSPTSR return code. Possible **rc** values are listed in the following table.

Table 155. rc values

Return code	Description
0	CIUSPTSR procedure that completed successfully.
4	CIUSPTSR procedure that completed successfully, but one or more SQL warning conditions were received.
8	CIUSPTSR procedure that failed because of a critical error that is caused by incorrect input parameter values.
12	CIUSPTSR procedure that failed because of a disastrous error that is caused by SQL Exception conditions during SQL statement execution.

The **sqlcode** parameter values depend on the return code and can be found in the following table.

Table 156. sqlcode values

Return code	sqlcode value
0	0
4	Shows sqlcode for the last statement that caused the SQL warning condition.
8	0
12	Shows sqlcode of the failed SQL statement.

errmsg contains message text that describes the error or warning:

- For **rc=4**, it contains the SQL message of the last SQL statement that caused the warning condition.
- For **rc=8**, it provides the invalid parameter value that caused the error.
- For **rc=12**, it provides SQL error message for the failed SQL statement.

Returned result sets

The CIUSPTSR stored procedure returns result sets. The number and structure of the result sets is dependent on the input parameters values. A result set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in a result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop as shown in the COBOL example for the CIUSPAPP stored procedure. The following information describes the result sets that are returned by the CIUSPTSR procedure.

List summary threadsafe information about specified program

To list summary threadsafe information about the specified program, you must set the input parameters as follows:

```

ctype      = 'PGMS'
collid     = collection ID
applid     = applid
qarg       = program name
cicslevel = CICS TS level

```

If you want to list summary threadsafe information for all existing collection IDs, set **collid** to %.

If you want to list summary threadsafe information for all existing **applids**, set **applid** to %.

If you want to list summary threadsafe information for all existing programs, set **qarg** to %

Set **cicslevel** either to a particular CICS TS version or set it to null to use the CICS TS version of the region on which the program was run and collected. Null is the default.

This call returns one result set with all columns of the CIU_THREADSafe_SUMMARY global temporary table.

List summary threadsafe information about programs that are started by the specified transaction

To list summary threadsafe information about the programs that were started by the specified transaction, you must set the input parameters as follows:

```
ctype      = 'TRNS'  
collid     = collection ID  
applid     = APPLID  
qarg       = program name  
cicslevel  = CICS TS level
```

If you want to list summary threadsafe information for all existing collection IDs, set **collid** to %.

If you want to list summary threadsafe information for all existing APPLIDs, set **applid** to %.

If you want to list summary threadsafe information for programs that were started under any of existing transaction set **qarg** to %.

Set **cicslevel** either to particular CICS TS version or set it to null to use CICS TS version of the region on which the program was run and collected. Null is the default.

This call returns one result set with all columns of the CIU_THREADSafe_DETAIL global temporary table.

List detailed threadsafe information about specified program

To list detailed threadsafe information about the specified program, you must set the input parameters as follows:

```
ctype      = 'DTLD'  
collid     = collection ID  
applid     = APPLID  
qarg       = program name  
cicslevel  = CICS TS level
```

Set COLLID to existing CICS IA resources collection ID. Set APPLID to existing CICS TS region APPLID.

Set QARG to existing collected program name.

Set CICSLEVEL either to particular CICS TS version or set it to null to use CICS TS version of the region on which the program was run and collected. Null is the default.

This call returns one result set with all columns of the CIU_THREADSafe_DETAIL global temporary table.

CIUSPAP1 Stored Procedure

When you use the CIUSPAP1 stored procedure you can view or delete resource information for a CICS TS application definition.

What is the CIUSPAP1 Stored Procedure?

Use the CIUSPAP1 is a DB2 Stored Procedure to carry out the following actions:

- List CICS TS Applications for which you have collected data.
- Retrieve or delete resources that are used by a specific TS application or operations that are associated with that application.

Syntax

You can invoke the CIUSPAP1 procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPAP1 (calltype, platform, appname, version1,  
version2, version3, message, return-code);
```

Procedure parameters

There are several input parameters that help you to manage CIUSPAP1 processing and several output parameters that inform about the process completion and errors, if any.

This table lists all CIUSPAP1 parameters.

Table 157. CIUSPAP1 parameters

Parameter name	Input/Output	Type	Description
calltype	INPUT	CHAR(4)	Type of call.
platform	INPUT	CHAR(64)	Specifies the platform name (this parameter is used for all calltypes other than LIST).
appname	INPUT	CHAR(64)	Specifies the application name (this parameter is used for all calltypes other than LIST).
version 1	INPUT	INTEGER	Specifies a Major Version of Application (this parameter is used for all calltypes other than LIST).
version 2	INPUT	INTEGER	Specifies a Minor Version of Application (this parameter is used for all calltypes other than LIST).
version 3	INPUT	INTEGER	Specifies a Micro Version of Application (this parameter is used for all calltypes other than LIST).
operation	INPUT	CHAR(64)	Specifies the operation name (this parameter is optional and can be used for all calltypes other than LIST).
message	OUTPUT	CHAR(300)	Specifies a message buffer that contains error information for a List call type, or the result of procedure execution for a Delete call type.
return-code	OUTPUT	INTEGER	Return code.

CIUSPAP1 INPUT parameters (calltype, appname)

The **calltype** parameter specifies queries that can be called by the application. Table **calltype** values lists all available queries and their description.

Table 158. **calltype** values

calltype value	Description
LIST	List of Applications for which CICS IA has collected data.
DEL	Deletes all resources that are associated with the Application.
RES	Lists all resources that are associated with the Application.
CICS	Lists all CICS resources that are associated with the Application.
DB2	Lists all DB2 resources that are associated with the Application.
IMS	Lists all IMS resources that are associated with the Application.
MQ	Lists all WebSphere MQ resources that are associated with the Application.
NAT	Lists all Natural resources that are associated with the Application.

CIUSPAP1 OUTPUT parameters (error-message, return-code)

The **return-code** parameter contains value of the CIUSPAP1 return code. Possible **return-code** values are listed in this table:

Table 159. **return-code** values

Return code	Description
0	CIUSPAP1 procedure has completed successfully.
4	Any SQL Exception or SQL Warning occurred, in this case ERROR-MESSAGE contains the message with explanation of exception/warning.
8	One of the following conditions exists: <div> calltype Is invalid, that is the value is not L/D, in this case ERROR-MESSAGE contains the following text: 'Invalid call type' </div> <div> appname Is invalid, that is the value is empty, in this case ERROR-MESSAGE contains the following text: 'Application Name must be specified' </div>

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**=8, it provides the following message: “Application name must be specified”.

Listing all CICS TS applications

To retrieve a list of all CICS TS Applications that you have collected data for, set the input parameters as follows:

```
CALLTYPE='LIST'
```

Set all other input parameters to null values.

This call returns one OPEN cursor which you use to FETCH the returned data by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor,

and then the SQL FETCH loop as shown in the COBOL example for “CIUSPAPP Stored Procedure” on page 407.

Resources used by applications

To retrieve a list of all resources used by a CICS TS Application, set the input parameters as follows:

- **calltype** = 'RES'
- Set the **applname** and version fields as required.

To list the resources by the operation within an Application, set the **Operation** field to the name of the operation; otherwise, leave it blank.

This call returns one 5 OPEN cursor which you use to FETCH the returned data by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and the SQL FETCH loop as shown in the COBOL example for “CIUSPAPP Stored Procedure” on page 407.

CIUSPPUR Stored Procedure

When you use the CIUSPPUR stored procedure you can access saved dependency data directly from your application.

What is the CIUSPPUR Stored Procedure?

CIUSPPUR is a DB2 Stored Procedure that acts as a dependency data query interface that can be called from a user application with the SQL CALL statement. CIUSPPUR optimizes the size of the CICS IA DB2 tables when they contain a large amount of unnecessary TSQ or ENQ related dependency records that have equal prefixes.

Syntax

You can call the CIUSPPUR procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPPUR (ptype, preobj, plength, pcddct, prc,  
               perrmsg );
```

Procedure parameters

There are several input parameters that help you to manage CIUSPPUR processing and several output parameters that inform about the process completion and errors, if any.

This table lists all CIUSPPUR parameters.

Table 160. CIUSPPUR parameters

Parameter name	Input/Output	Type	Description
ptype	INPUT	CHAR(3)	Type of object.
preobj	INPUT	<=CHAR(100)	Prefix of OBJECT fields.
plength	INPUT	INTEGER	Length of preobj parameter.
pcddct	OUTPUT	INTEGER	Count of deleted rows in the CIU_CICS_DATA table.
prc	OUTPUT	INTEGER	Return code.
perrmsg	OUTPUT	CHAR(300)	Error message text.

CIUSPPUR INPUT parameters (ptype, preobj, plength)

The **ptype** parameter specifies the resource type, and has two values: TSQ or ENQ. The **ptype** parameter defines which cursors the CIUSPPUR program opens.

The **preobj** parameter is a prefix of the similar TSQ or ENQ resource names in the CIU_CICS_DATA table for which you want to make a DB2 compression.

The **plength** parameter specifies the prefix length. The valid values of the **plength** parameter are in the range from 1 - 100, specified automatically by a call from the JCL- COBOL programs.

CIUSPPUR OUTPUT parameters (pcddct, prc, permsg)

The **pcddct** parameter contains the number of deleted rows in the CIU_CICS_DATA table with the duplicate combination of **type**, **function**, **program**, **offset**, **proglen** and **prefix** of resource.

The **prc** parameter contains the value of the CIUSPPUR return code. Possible **prc** values are listed in this table:

Table 161. prc values

Return code	Description
0	Procedure has completed successfully.
8	Procedure failed because of a critical error caused by an incorrect parameter value.
12	Procedure failed because of a disaster error caused by SQL Exception conditions during SQL statement execution.

The **permsg** parameter contains message text that describes the error or warning:

- For **prc**=8, it provides the following message: "Invalid type".
- For **prc**=12, it provides an SQL error message for the failed SQL statement.

Returned result sets

Procedure CIUSPPUR does not return result sets because it directly modifies the contents of database tables.

CIUSPPUR invocation

An example of a COBOL program, which calls the CIUSPPUR stored procedure:

```
IDENTIFICATION DIVISION.  
    PROGRAM-ID.    CIUDBPUR  
    EJECT  
    ENVIRONMENT DIVISION.  
    CONFIGURATION SECTION.  
    SOURCE-COMPUTER.    IBM3090.  
    OBJECT-COMPUTER.    IBM3090.  
    INPUT-OUTPUT SECTION.  
    FILE-CONTROL.  
    DATA DIVISION.  
    FILE SECTION.  
    EJECT  
    WORKING-STORAGE SECTION.  
    *****  
    *  
    *    WORKING VARIABLES
```



```

*
*****
01 WS-VARIABLES.
   03 WS-IN-TYPE          PIC X(3).
   03 WS-IN-PREOBJ        PIC X(100).
   03 WS-IN-PREOBJ-LEN    PIC S9(8) BINARY.
   03 WV-PTYPE            PIC X(3).
   03 WV-PREOBJ           PIC X(100).
   03 WV-LENGTH           PIC S9(8) BINARY.
   03 WV-PCDDCT           PIC S9(8) BINARY.
   03 WV-ERRMSG           PIC X(300).
   03 WV-RC               PIC S9(8) BINARY.

*
01 WS-PARSED-PARAMETERS.
   03 WS-LENGTH           PIC S9(8) COMP.
   03 WS-PARSED-COMMA-COUNT PIC S999 COMP VALUE 0.
   03 WS-INPUT-PARM       PIC X(100).
* 'type='
   03 WS-PARSED-P1.
      05 WS-PARSED-P1-VALUE PIC X(4) VALUE SPACES.
      05 WS-PARSED-P1-LENGTH PIC S999 COMP VALUE 0.
      05 WS-PARSED-DELIMITER1 PIC X VALUE SPACES.
* 'ENQ,' or 'TSQ,'
   03 WS-PARSED-P2.
      05 WS-PARSED-P2-VALUE PIC X(3) VALUE SPACES.
      05 WS-PARSED-P2-LENGTH PIC S999 COMP VALUE 0.
      05 WS-PARSED-DELIMITER2 PIC X VALUE SPACES.
   03 WS-PARSED-P3.
* 'prefix='
      05 WS-PARSED-P3-VALUE PIC X(6) VALUE SPACES.
      05 WS-PARSED-P3-LENGTH PIC S999 COMP VALUE 0.
      05 WS-PARSED-DELIMITER3 PIC X VALUE SPACES.
   03 WS-PARSED-P4.
* 'XXXXXXXXXXXXXXXX..XX' - prefix
      05 WS-PARSED-P4-VALUE PIC X(100) VALUE SPACES.
      05 WS-PARSED-P4-LENGTH PIC S999 COMP VALUE 0.
      05 WS-PARSED-DELIMITER4 PIC X VALUE SPACES.
*****
*
*   CONSTANT VALUES
*
*****
01 WC-CONSTANTS.
   03 WC-COMMA          PIC X VALUE ',' .
   03 WC-EQUATION       PIC X VALUE '=' .

*****
*
*   MESSAGE AREA
*
*****

01 WM-MESSAGES.
   03 WM-MESSAGE-0.
      05 WM-MSGNUMB PIC S9(8) BINARY VALUE 9999.
      05 WM-SUBNUMB1 PIC S9(8) BINARY VALUE 1.

EJECT
*****
*
*   DB2 COMMUNICATIONS AREA
*
*****

***   SQL COMMUNICATIONS AREA

EXEC SQL

```

```

        INCLUDE SQLCA
    END-EXEC.

01  WK-END                                PIC X(45) VALUE
    '*** END OF WORKING STORAGE FOR CIUDBPUR ***'.

EJECT

LINKAGE SECTION.
01  PARAMETER.
    05  P-LENGTH                        PIC S999 COMP.
    05  INPUT-PARMS                     PIC X(116).

PROCEDURE DIVISION USING PARAMETER.
VALIDATE-PARAMETERS SECTION.

*** CHECK LENGTH OF INPUT STRING
    IF ((P-LENGTH < 17) OR (P-LENGTH > LENGTH OF INPUT-PARMS))
    THEN
        MOVE 6062 TO WM-MSGNUMB
        MOVE P-LENGTH TO WS-LENGTH
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 INPUT-PARMS WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-IF.

*** COUNT NUMBER OF PARAMETERS (1 comma => 2 parameters)
    INSPECT INPUT-PARMS(1:P-LENGTH)
        TALLYING WS-PARSED-COMMA-COUNT
        FOR ALL WC-COMMA.

*** PARSE PARAMETERS INTO STRUCTURE
    UNSTRING INPUT-PARMS(1:P-LENGTH)
    DELIMITED BY WC-COMMA OR WC-EQUATION
    INTO WS-PARSED-P1-VALUE DELIMITER IN WS-PARSED-DELIMITER1
    COUNT IN WS-PARSED-P1-LENGTH
    WS-PARSED-P2-VALUE DELIMITER IN WS-PARSED-DELIMITER2
    COUNT IN WS-PARSED-P2-LENGTH
    WS-PARSED-P3-VALUE DELIMITER IN WS-PARSED-DELIMITER3
    COUNT IN WS-PARSED-P3-LENGTH
    WS-PARSED-P4-VALUE DELIMITER IN WS-PARSED-DELIMITER4
    COUNT IN WS-PARSED-P4-LENGTH
    ON OVERFLOW
        MOVE 6062 TO WM-MSGNUMB
        MOVE P-LENGTH TO WS-LENGTH
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 INPUT-PARMS WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-UNSTRING.

*** CHECK PARAMETERS
    EVALUATE WS-PARSED-COMMA-COUNT
    WHEN 0
        MOVE 6062 TO WM-MSGNUMB
        MOVE P-LENGTH TO WS-LENGTH
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 INPUT-PARMS WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    WHEN 1
        IF (WS-PARSED-P1-VALUE(1:4) NOT = 'TYPE') OR
            (WS-PARSED-DELIMITER1 NOT = WC-EQUATION) OR
            (WS-PARSED-P1-LENGTH NOT = 4)
        THEN
            MOVE 6062 TO WM-MSGNUMB

```

```

        MOVE WS-PARSED-P1-LENGTH TO WS-LENGTH
        MOVE WS-PARSED-P1-VALUE TO WS-INPUT-PARM
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 WS-INPUT-PARM WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-IF
    IF ((WS-PARSED-P2-VALUE(1:3) NOT = 'TSQ') AND
        (WS-PARSED-P2-VALUE(1:3) NOT = 'ENQ')) OR
        (WS-PARSED-DELIMITER2 NOT = WC-COMMA ) OR
        (WS-PARSED-P2-LENGTH NOT = 3)
    THEN
        MOVE 6062 TO WM-MSGNUMB
        MOVE WS-PARSED-P2-LENGTH TO WS-LENGTH
        MOVE WS-PARSED-P2-VALUE TO WS-INPUT-PARM
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 WS-INPUT-PARM WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-IF
    MOVE WS-PARSED-P2-VALUE(1:3) TO WS-IN-TYPE
    IF (WS-PARSED-P3-VALUE(1:6) NOT = 'PREFIX') OR
        (WS-PARSED-DELIMITER3 NOT =WC-EQUATION ) OR
        (WS-PARSED-P3-LENGTH NOT = 6)
    THEN
        MOVE 6062 TO WM-MSGNUMB
        MOVE WS-PARSED-P3-LENGTH TO WS-LENGTH
        MOVE WS-PARSED-P3-VALUE TO WS-INPUT-PARM
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 WS-INPUT-PARM WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-IF
    IF (WS-PARSED-P4-LENGTH = 0 OR WS-PARSED-P4-LENGTH >
        LENGTH OF WS-IN-PREOBJ)
    THEN
        MOVE 6062 TO WM-MSGNUMB
        MOVE WS-PARSED-P4-LENGTH TO WS-LENGTH
        MOVE WS-PARSED-P4-VALUE TO WS-INPUT-PARM
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 WS-INPUT-PARM WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-IF
    MOVE WS-PARSED-P4-VALUE(1:LENGTH OF WS-IN-PREOBJ)
        TO WS-IN-PREOBJ
    MOVE WS-PARSED-P4-LENGTH TO WS-IN-PREOBJ-LEN
    WHEN OTHER
        MOVE 6062 TO WM-MSGNUMB
        MOVE P-LENGTH TO WS-LENGTH
        CALL 'CIUCMSG' USING WM-MSGNUMB
            WM-SUBNUMB1 INPUT-PARMS WS-LENGTH
        MOVE 8 TO RETURN-CODE
        STOP RUN
    END-EVALUATE.

```

EJECT

A-MAIN SECTION.

```

        PERFORM B-INITIALIZE
        PERFORM C-MAIN-PROCESS
        PERFORM D-TERMINATE

```

A-EXIT.

STOP RUN.

EJECT

```

B-INITIALIZE SECTION.
*****
* THIS SECTION OPENS THE FILES
*****
B-INIT.

        MOVE SPACES          TO WV-ERRMSG

        MOVE ZEROS           TO WV-RC
                                WV-PCDDCT
B-EXIT.
EXIT
.
EJECT

C-MAIN-PROCESS SECTION.
*****
C-START.
        MOVE WS-IN-TYPE TO WV-PTYPE
        MOVE WS-IN-PREOBJ TO WV-PREOBJ
        MOVE WS-IN-PREOBJ-LEN TO WV-LENGTH

        EXEC SQL CALL CIUSPPUR(:WV-PTYPE,
                                :WV-PREOBJ,
                                :WV-LENGTH,
                                :WV-PCDDCT,
                                :WV-RC,
                                :WV-ERRMSG)

        END-EXEC

.
C-EXIT.
EXIT
.
EJECT

D-TERMINATE SECTION.
*****
* THIS SECTION CLOSES THE FILES AND OUTPUTS DETAILS.
*****
D-START.

        IF WV-RC = 0 THEN
            DISPLAY WV-ERRMSG
            DISPLAY 'DELETE REPORT FOR 'WV-PTYPE' TYPE and '
            WV-PREOBJ' prefix:'
            DISPLAY 'The number of deleted rows in the '
            DISPLAY 'CIU_CICS_DATA table = ' WV-PCDDCT
            DISPLAY 'CIU_RESOURCE table should be updated.'
        ELSE
            MOVE WV-RC TO RETURN-CODE
            DISPLAY WV-ERRMSG
        END-IF

.
D-EXIT.
EXIT
.
EJECT
.

```

Stored Procedures for Application Deployment

In CICS IA V5.3, there are four new stored procedures that are currently used exclusively by the CICS IA Explorer plug-in.

The new stored procedures are provided to assist you to identify:

- Your CICS Application entry points
- Your CICS Application Dependencies

A short description is provided for these new stored procedures. You need to obtain this information when you start to deploy the new “Platform and Application” concepts that are introduced in CICS IA V5.1, onwards. These stored procedures are shipped in the NATIVE format only and can be used only if your CICS IA database is using DB2 V9.1, or later.

CIUSPEPS Stored Procedure

CIUSPAPP is a stored procedure that you can use to identify and group your application entry points by the resource used. The type of resource that is used can be any CICS, DB2, MQ or IMS resource.

For example, CIUSPEPS can show you the possible entry points for all of the transactions that use a CICS FILE called PAYROLL.

Syntax

You can call the CIUSPEPS procedure with the following SQL CALL statement:

```
EXEC SQL
CALL CIUSPEPS (in_calltype, in_collid, in_applid, in_object, in_objtype1, in_objtype2,
               prc, psqlcode, psqlstate, permsg);
```

The following example shows what might be passed in the SQL CALL:

```
EXEC SQL
CALL CIUSPEPS ('T','%','%','PAYROLL','CICS','FILE', prc, psqlcode, psqlstate, permsg);
```

Note: The “%” are generic and return a result set for all of the collection IDs and applids.

Procedure parameters

There are several input parameters that help you to manage the CIUSPEPS processing and several output parameters that provide information about the process completion and any errors.

The following table lists all the CIUSPEPS parameters.

Table 162. CIUSPEPS parameters

Parameter name	input/output	Type	Description
in_calltype	INPUT	CHAR(1)	Type of call T or P for transaction and program entry points. U for Webservice type entry points (URIMAP). A for all entry points.
in_collid	INPUT	CHAR(16)	Specifies the collection ID for which you want to list the entry points.
in_applid	INPUT	CHAR(8)	Specifies the application ID for which you want to list the entry points.
in_object	INPUT	CHAR(255)	Specifies the name of the resource object for which you want to group your entry points.

Table 162. CIUSPEPS parameters (continued)

Parameter name	input/output	Type	Description
in_objtype1	INPUT	CHAR(4)	Specifies the main type of resource object on which you want to group your entry points. The type can be CICS, DB2, IMS, or MQ.
in_objtype2	INPUT	CHAR(16)	Specifies the secondary type of resource object on which you want to group your entry points. For example, for DB2 it might be TABLE.
prc	OUTPUT	INTEGER	Return Code
psqlcode	OUTPUT	INTEGER	DB2 SQL code
psqlstate	OUTPUT	CHAR(5)	DB2 SQL state
perrmsg	OUTPUT	VARCHAR(300)	Error message

Returned result sets

The CIUSPEPS stored procedure returns one or more result sets. The structure of the result sets is dependent on the input parameters values. A result set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in a result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop.

If you select ALL you get 2 results sets returned. One for the Transaction or Program and one for the Webservice .

For both TRANSACION and PROGRAM type entry points (T or P) it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, APPLID, PLATFORM, APPL_NAME, APL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, TRANSID , INITIAL_PROGRAM, and BACK_PROGRAM.

In this case, the entry points can be based on the TRANSACTION , the INITIAL_PROGRAM, or the BACK_PROGRAM.

CICS TS V5.1 supports only **PROGRAM** entry points.

CICS TS V5.3 supports **TRANSACTION** entry points.

For both Webservice type entry points (U) it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, APPLID, PLATFORM, APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, NAME, PROGRAM, URIMAP, and BACK_PROGRAM.

In this case the entry points can be based on the **PROGRAM, URIMAP, or the BACK_PROGRAM.**

CICS TS V5.1 supports only **PROGRAM** entry points.

CICS TS V5.2 supports **URIMAP** entry points.

CIUSPEP2 Stored Procedure

CIUSPEP2 is a stored procedure that you can use to identify your application entry points for a transaction, program, or web service and group them by the type of the resources they use.

For example, CIUSPEP2 can show you the possible entry points for transaction SSC1 and all other transactions that use MAPSETs, which are used by transaction SSC1.

Syntax

You can call the CIUSPEP2 procedure with the following SQL CALL statement:

```
EXEC SQL  
CALL CIUSPEP2 (in_calltype, in_collid, in_applid, in_object, in_objtype1, in_objtype2,  
               prc, psqlcode, psqlstate, permsg);
```

The following example shows what might be passed in the SQL CALL:

```
EXEC SQL  
CALL CIUSPEP2 ('T','%','%','SSC1','CICS','FILE','MAPSET', prc, psqlcode, psqlstate, permsg);
```

Note: The “%” symbols are generic and return a result set for all of the collection IDs and applids.

Procedure parameters

There are several input parameters that help you to manage CIUSPEP2 processing and several output parameters that inform you about the process completion and any errors.

The following table lists all of the CIUSPEP2 parameters.

Table 163. CIUSPEP2 parameters

Parameter name	input/output	Type	Description
in_calltype	INPUT	CHAR(1)	Type of call T or P for transaction and program entry points. U for Webservice type entry points (URIMAP).
in_collid	INPUT	CHAR(16)	Specifies the collection ID for which you want to list the entry points.
in_applid	INPUT	CHAR(8)	Specifies the application ID for which you want to list the entry points.
in_object	INPUT	CHAR(255)	Specifies the name of the resource object for which you want to group your entry points.
in_objtype1	INPUT	CHAR(4)	Specifies the main type of resource object on which you want to group your entry points. The type can be CICS, DB2, IMS, or MQ.
in_objtype2	INPUT	CHAR(16)	Specifies the secondary type of resource object on which you want to group your entry points.
prc	OUTPUT	INTEGER	Return Code
psqlcode	OUTPUT	INTEGER	DB2 SQL code
psqlstate	OUTPUT	CHAR(5)	DB2 SQL state
permsg	OUTPUT	VARCHAR(300)	Error message

Input parameter **in_objtype2** is restricted to the values in table xyz and is dependent on the value of input parameter **in_objtype1**.

Table 164. Secondary resource type

in_objtype1	
CICS	FILE , MAPSET , MAP, TDQUEUE, TSQUEUE, CONTAINER, CHANNEL
DB2	TABLE , VIEW , CURSOR
MQ	QUEUE
IMS	PCB, PSB

Returned result sets

The CIUSPEP2 stored procedure returns one or more result sets. The structure of the result sets is dependent on the input parameters values. A result set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in a result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop.

For both TRANSACION and PROGRAM type entry points (T or P) it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, **APPLID**, **PLATFORM**, **APPL_NAME**, **APPL_VER1**, **APPL_VER2**, **APPL_VER3**, **APPL_OPER**, **TRANSID** , **INITIAL_PROGRAM**, **BACK_PROGRAM**, and **OBJECT**.

In this case, the entry points can be based on the **TRANSACTION** , the **INITIAL_PROGRAM**, or the **BACK_PROGRAM**.

Note: For DB2 we return the **RESNAME** column and not the **OBJECT** column.

For both Webservice type entry points (U) it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, **APPLID**, **PLATFORM**, **APPL_NAME**, **APPL_VER1**, **APPL_VER2**, **APPL_VER3**, **APPL_OPER**, **NAME** , **PROGRAM**, **URIMAP**, **BACK_PROGRAM**, and **OBJECT**.

In this case the entry points can be based on the **PROGRAM**, **URIMAP**, or the **BACK_PROGRAM**.

CIUSPEP3 Stored Procedure

CIUSPEP3 is a stored procedure that you can use to identify your application entry points for an existing platform, application, or Operation.

For example, CIUSPEP3 can show you all of the possible entry points for the TEST_OPERATIONS application. This is the DUMMY application name that is used by CICS IA to store application and operation names that are captured as a result of editing the PROGRAM resource and defining an operation name.

Syntax

You can call the CIUSPEP3 procedure with the following SQL CALL statement:


```
EXEC SQL
CALL CIUSPEP3 (in_calltype, in_collid, in_applid, in_platform, in_aplname, in_applver1,
               in_applver2, in_applver3, in_apploper, prc, psqlcode, psqlstate, perrmsg);
```

The following example shows what might be passed in the SQL CALL:

```
EXEC SQL
CALL CIUSPEP2 ('A','%',' ','TEST_OPERATIONS','0','0','0',' ',prc, psqlcode, psqlstate, perrmsg);
```

Procedure parameters

Several input parameters can help you to manage CIUSPEP3 processing and several output parameters that inform about the process completion and errors, if any.

The following table lists all of the CIUSPEP3 parameters.

Table 165. CIUSPEP3 parameters

Parameter name	Input/output	Type	Description
in_calltype	INPUT	CHAR(1)	Type of call. P for a list of entry points by platform. A for a list of entry points by application. 0 for a list of entry points by operation.
in_collid	INPUT	CHAR(16)	Specifies the collection ID for which you want to list the entry points.
in_platform	INPUT	CHAR(64)	Specifies the platform name. Required for P, A, and 0. Note: It can be blank for A or O but must not be NULL.
in_aplname	INPUT	CHAR(64)	Specifies the application name. Required for call types A and 0.
in_applver1	INPUT	INTEGER	Specifies application MAJOR version. Required for call types A and 0.
in_applver2	INPUT	INTEGER	Specifies application MINOR version. Required for call types A and 0.
in_applver3	INPUT	INTEGER	Specifies application MICRO version. Required for call types A and 0.
in_apploper	INPUT	INTEGER	Specifies operation version. Required for call type 0.
prc	OUTPUT	INTEGER	Return Code
psqlcode	OUTPUT	INTEGER	DB2 SQL code
psqlstate	OUTPUT	CHAR(5)	DB2 SQL state
perrmsg	OUTPUT	VARCHAR(300)	Error message

Returned result sets

The CIUSPEP3 stored procedure returns two result sets for all types of call types. The structure of the result sets depends on the input parameters values. A result

set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in a result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop.

For PLATFORM, APPLICATION and OPERATION call types return 2 result sets. The first set contains the TRANSID or PROGRAM entry points and it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, APPLID, PLATFORM, APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, TRANSID, INITIAL_PROGRAM, and BACK_PROGRAM.

In this case the entry points can be based on the TRANSACTION, the INITIAL_PROGRAM, or the BACK_PROGRAM.

The second set contains the web service type entry points and it returns a result set, which contains the following DB2 columns:

COLLECTION_ID, APPLID, PLATFORM, APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, NAME, PROGRAM, URIMAP, and BACK_PROGRAM.

CIUSPDEP Stored Procedure

CIUSPDEP is a stored procedure that returns a list of application dependencies, which can be added to a CICS Bundle that is then used to define a CICS Application. CIUSPDEP can be called against a PLATFORM, an APPLICATION, or an OPERATION. It returns information about your CICS, DB2, and MQ dependencies.

For example, CIUSPDEP can show you all of the possible dependencies for the TEST_OPERATIONS application. The TEST_OPERATIONS application is the DUMMY application name that is used by CICS IA to store application and operation names that are captured as a result of editing the PROGRAM resource and defining an operation name.

Syntax

You can call the CIUSPDEP procedure with the following SQL CALL statement:

```
EXEC SQL
CALL CIUSPDEP (in_calltype, in_collid, in_applid, in_platform, in_aplname, in_applver1,
               in_applver2, in_applver3, in_apploper, prc, psqlcode, psqlstate, perrmsg);
```

The following example shows what might be passed in the SQL CALL:

```
EXEC SQL
CALL CIUSPDEP ('A','%',' ','TEST_OPERATIONS','0','0','0',' ',prc, psqlcode, psqlstate, perrmsg);
```

Procedure parameters

Several input parameters can help you to manage CIUSPEP3 processing and several output parameters that inform about the process completion and errors, if any.

The following table lists all of the CIUSPDEP parameters.

Table 166. CIUSPDEP parameters

Parameter name	Input/output	Type	Description
in_calltype	INPUT	CHAR(1)	Type of call. P for a list of dependencies by platform. A for a list of dependencies by application. 0 for a list of dependencies by operation.
in_collid	INPUT	CHAR(16)	Specifies the collection ID for which you want to list the dependencies.
in_platform	INPUT	CHAR(64)	Specifies the platform name. Required for P, A, and 0. Note: It can be blank for A or O but must not be NULL.
in_applname	INPUT	CHAR(64)	Specifies the application name. Required for call types A and 0.
in_applver1	INPUT	INTEGER	Specifies application MAJOR version. Required for call types A and 0.
in_applver2	INPUT	INTEGER	Specifies application MINOR version. Required for call types A and 0.
in_applver3	INPUT	INTEGER	Specifies application MICRO version. Required for call types A and 0.
in_apploper	INPUT	INTEGER	Specifies operation version. Required for call type 0.
prc	OUTPUT	INTEGER	Return Code
psqlcode	OUTPUT	INTEGER	DB2 SQL code
psqlstate	OUTPUT	CHAR(5)	DB2 SQL state
perrmsg	OUTPUT	VARCHAR(300)	Error message

Returned result sets

The CIUSPDEP stored procedure returns five result sets for all types of call types. The structure of the result sets depends on the input parameters values. A result set is a set of rows that are associated with a cursor opened in the stored procedure and returned to caller program. A result set is effectively a table. You can access the data that is returned in a result set by running an SQL ASSOCIATE LOCATORS, followed by an SQL ALLOCATE cursor, and then the SQL FETCH loop.

RESULT SET 1

'CICS DEPENDENCIES BY...',**COLLECTION_ID**, **APPLID**, **PLATFORM**, **APPL_NAME**, **APPL_VER1**, **APPL_VER2**, **APPL_VER3**, **APPL_OPER**, **TRANSID**, "PROGRAM", "TYPE",**OBJECT**

In this case, the dependency TYPE is described by the TYPE column and the dependency name is in the OBJECT column.

RESULT SET 2

'CICS LIBRARY DEPENDENCIES BY...','COLLECTION_ID, APPLID, PLATFORM,
APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, PROGRAM_NAME,
LIB_NAME,LIB_DATASET_NAME

In this case, the dependency is the LIB_NAME and the type is LIBRARY.

RESULT SET 3

'CICS EVENT DEPENDENCIES BY...','COLLECTION_ID, APPLID, PLATFORM, APPL_NAME,
APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, EVENT_NAME, EVENT_BINDING,EB_ADAPTER

In this case, the dependency name can be either the EVENT_BINDING, or the
EB_ADAPTER, or both of them.

RESULT SET 4

'DB2 CONNECTION DEPENDENCY BY...','COLLECTION_ID, APPLID, PLATFORM,
APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER, DB2ID

In this case, you are informed that you have a dependency on a DB2 connection
definition. You are given the subsystem ID but not the resource definition name.

RESULT SET 5

'MQ CONNECTION DEPENDENCY BY...','COLLECTION_ID, APPLID, PLATFORM,
APPL_NAME, APPL_VER1, APPL_VER2, APPL_VER3, APPL_OPER

In this case, you are informed that there is a dependency on an MQ connection
definition. The subsystem ID and the resource definition name are not captured.
However, you did use an MQ resource.

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 controls 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 XEIOU 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 assembly 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 following table:

Table 167. 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 Command Flow Options panel, CIUA01.

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 63 on page 441.

```

*****
EISPLI EQU X'02'
EISCOBOL EQU X'04'
EISASM EQU X'08'
COPY DFHEIPDS
R0 EQU 0
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
R9 EQU 9
RA EQU 10
RB EQU 11
RC EQU 12
RD EQU 13
RE EQU 14
RF EQU 15
EJECT
CIUUESMP CSECT
CIUUESMP AMODE 31
CIUUESMP RMODE ANY
SPACE
SAVE (14,12)
BALR RC,0
USING *,RC
LR R5,R1
LR RB,RD
USING CIUEPAR,R5
L RD,MT_UE_WKAA
ST RB,4(,RD)
SPACE
DROP R1
L R6,MT_UE_ARG1
USING EIA,R6
L R7,EIAARG0
USING EID,R7
SPACE
CLC EIDFN,LINKID
JNE SKIPADD
L R2,MT_UE_PGNA
*
LA R3,PROGNAME
CLC 0(8,R2),0(R3)
*
JNE SKIPADD
L R2,MT_UE_DAT1A
MVC 0(19,R2),UDATA1
L R3,EIAARG1
MVC 11(8,R2),0(R3)
L R2,MT_UE_DAT2A
MVC 0(9,R2),UDATA2
CLI EIDOPT2,EIDCOMM
JNE NOCOMMA
L R3,EIAARG2
MVC 11(22,R2),0(R3)
J PUTRC0
NOCOMMA MVC 9(11,R2),NOCOMM
PUTRC0 L R2,MT_UE_RETCA
L R3,RC0
ST R3,0(,R2)
J ENDPOINT
SKIPADD L R2,MT_UE_RETCA
L R3,RC8
ST R3,0(,R2)
ENDPOINT L RD,4(,RD)
RETURN (14,12)
*
EJECT
*
RETURN CODES:

```

PARAMETER REGISTER

CIUEPAR Base Register

EXEC CICS ARG List Base Register

EXEC CICS ARG0 Base Register

BASE

SAVE AREA

RETURN ADDRESS

ENTRY ADDRESS

SAVE REGISTERS

LOAD BASE REGISTER

LOAD PARAMETER LIST ADDRESS TO R5

BASE USER EXIT PARMLIST

LOAD WORK AREA ADDRESS TO R13

SAVE OLD SAVE AREA ADDRESS IN WORK AREA

EIA WAS BASED ON R1 IN COPYBOOK

LOAD ADDRESS OF EXEC CICS ARGS LIST

BASE ADDRESSES OF EXEC CICS ARGS

LOAD ARG0 ADDRESS

BASE ARG0

LINK COMMAND?

NO, LEAVE

LOAD ADDRESS OF PROGRAM NAME, FROM

WHICH EXEC CICS LINK GIVES CONTROL

LOAD ADDRESS OF REQUIRED PROGRAM NAME

COMPARE CURRENT PROGRAM NAME WITH

PROGRAM NAME WE ARE INTERESTED IN.

LEAVE IF IT IS NOT THE DESIRED PROGRAM

COPY STRING TO USER DATA FIELD ONE

ADDRESS OF ARG1 - CALLED PROGRAM NAME

COPY IT TO USER DATA

COPY STRING TO USER DATA FIELD TWO

DOES COMMAREA EXIST?

NO, WRITE THAT IT DOES NOT EXIST

ADDRESS OF ARG2 - PASSED COMMAREA

PUT 22 BYTES FROM COMMAREA TO CIUUDAT2

COPY STRING TO USER DATA FIELD TWO

SAVE RETURN CODE RC=0

SAVE RETURN CODE RC=8

RESTORE OLD SAVE AREA ADDRESS

RESTORE REGISTERS AND

RETURN TO CALLER

Appendix H. Collecting dynamic COBOL calls

CICS IA detects dynamic COBOL calls, which are collected by the Dependency and Command Flow Data collectors.

To support the collection of dynamic COBOL Language Environment calls, CICS IA assumes that the Call parameter list for IBM COBOL for OS/390[®] program conforms to the structure documented in the publication *Language Environment Vendor Interfaces for COBOL Call routine*

To support the collection of dynamic VS COBOL II calls, the call must be an ID type call and not a Literal call.

If neither of the two previous statements applies, the called program name is not collected and the called name is replaced in the **OBJECT** field:

UNEXP-hn

where *n* is reserved for IBM diagnostics,

and *h* is a hex digit that indicates the program attributes as shown:

- | | |
|-------------|---|
| 0xxx | The calling program is detected as COBOL for OS/390. |
| 1xxy | The calling program is detected as VS COBOL II program. |
| x0xx | The called program is detected as COBOL for OS/390. |
| x1xx | The called program is detected as VS COBOL II program. |
| xx0x | The calling program AMODE is detected as 31. |
| xx1x | The calling program AMODE is detected as 24. |
| xxx0 | The AMODE of the called program is detected as 31 bit. |
| xxx1 | The AMODE of the called program is detected as 24 bit. |

Appendix I. 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 these steps:

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

Special characters

hlqc.CICS.CSV file 83

A

abend codes 319
accessibility 445
activity, affinity lifetimes 9
add user menu 89
affinities
 collecting data 20
 commands detected 239
 functions 6
 related CICS API commands 12
 related CICS SPI commands 12
Affinities Reporter 169, 174, 176
 affinity transaction groups,
 modifying 175
 CIUAFFRD job 170
 CIUAFFRP job 170
 output 170
 output report (example) 171
 overview 25
 running 169
 understanding the affinities 175
affinities, understanding them 175
affinity
 inter-transaction 8
 transaction-system 8
affinity database objects
 overview 23
Affinity database objects
 updating 137
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 269
Affinity objects
 base tables 269
 CIU_AFF_CMD_DATA table 271
 CIU_AFF_INDEX_DATA table 271
 facilitating tables 271
 structure 269
 V_CIU_AFFINITY view 272
affinity relations 8
 BAPPL 8
 global 8
 LINK3270 8
 LUName 8
 user ID 8
affinity tables and indexes
 CICS region tables 403
 CSECT Module Scanner tables 402

affinity tables and indexes (*continued*)
 DB2 space allocation 400
 Load Module Scanner tables 401
affinity transaction group definitions,
 producing 173
affinity transaction-groups,
 combining 202
affinity views 272
affinity-related commands detected by
 the Collector 239
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 219
 presentation commands 219
application data 115
ASMTDLI commands detected by the
 Collector 238
assembler language, example, Load
 Module Scanner 247

B

BAPPL, affinity relations 8
base table, CICS regions 280
base tables
 Load Module Scanner 275
basic affinity transaction-groups,
 combining 202
batch data interchange commands
 presentation commands 219
BMS commands 218
builder
 overview 27
Builder 197
 changes to CIUAFFBL job 197
 combined affinity transaction group
 definitions 201
 combining basic affinity transaction
 groups 202
 data sets processed report 203
 empty transaction groups report 204
 error report 205
 group merge report 204
 HEADER statements 200
 how to run 197
 input parameters 197
 output 201
 syntax for input to 198

C

calculating CICS data space
 space considerations 389
calculating DB2 data space
 space considerations 389
calculating IMS data space
 space considerations 389
calculating MQ data space
 space considerations 390
calculating Natural data space
 space considerations 390
calculating resource data space
 space considerations 390
Calculating the space required for
 Interdependency Collection
 space considerations 388
calculation for the space required for
 affinity collection
 space considerations 391
CBLTDLI commands detected by the
 Collector 238
CFDTPPOOL 229
changes for Version 3 Release 1 xxxv
changes for Version 3 Release 2 xxviii
changes for Version 5 Release 1 xxiii
changes for Version 5 Release 2 xix
changes for Version 5 Release 3 xv
changing Collector options 105
Changing the options of the collection of
 data 97
CICS API commands 218
CICS API commands detected
 business transaction services (BTS)
 commands 240
 interval control commands 239
 other commands 240
 program control commands 239
 storage control commands 239
 task control commands 239
 temporary storage commands 240
CICS commands detected by the
 Collector 217
CICS configuration
 prepare your environment 31
CICS Explorer plug-in variables 56
CICS FEPI commands detected 236
 CONNECTION 236
 NODE 236
 POOL 236
 PROPERTYSET 236
 TARGET 237
cics ia command flow application data
 collection panel, displaying 130
cics ia command flow applid list panel,
 displaying 129
cics ia command flow statistics 131
CICS IA command flow variables 53
CICS IA plug-in
 hover help 445
CICS IA plug-in level 213
CICS IA security 405

- CICS IA UDB database 151
- CICS IA variables 52
- CICS region tables
 - affinity tables and indexes 403
- CICS regions base table 280
- CICS regions objects
 - CIU_REGION_INFO table 280
 - structure 280
- CICS resource objects
 - CIU_CONNECTIONS table 281
 - CIU_EXIT_INFO table 293
 - CIU_FILE_DETAIL table 282
 - CIU_PROGRAM_DETAIL table 284
 - CIU_TDQUEUE_DETAIL table 289
 - CIU_TRUEEXIT_INFO table 294
 - CIU_TSQUEUE_DETAIL table 291
 - V_CIU_TRUEEXIT_INFO table 294
- CICS resource objectsCICS resource
 - objects
 - GLUE and TRUE exit resource
 - table 293
- CICS SPI commands detected 225, 240
 - AUTOINSTALL 225
 - BRFACILITY 225
 - CAPTURE 226
 - CONNECTION 226, 230, 231
 - CORBASERVER 226
 - CREATE MAPSET 226
 - CREATE PARTITIONSET 226
 - CREATE SESSIONS 226
 - CREATE TYPETERM 226
 - DB2CONN 227
 - DELETSHIPED 228
 - DISPATCHER 228
 - DJAR 228
 - DOCTEMPLATE 228
 - DSNAME 228
 - DUMPDS 228
 - ENQMODEL 228
 - EPADAPTER 228
 - EXIT 229
 - FILE 229
 - HOST 229
 - INQUIRE 229
 - IPCONN 230
 - JOURNALMODEL 230
 - JOURNALNAME 230
 - JOURNALNUM 230
 - LIBRARY 231
 - LSRPOOL 226
 - MONITOR 231
 - NETNAME 231
 - PARTNER 231
 - PIPELINE 232
 - PROCESSTYPE 231
 - PROFILE 232
 - PROGRAM 232
 - RESYNC 232
 - SYSDUMPCODE 232
 - SYSTEM 232
 - TASK 233
 - TCLASS 233
 - TCPIP 233
 - TCPIPSERVICE 233
 - TDQUEUE 233
 - TEMPSTORAGE 233
 - TERMINAL 232, 233
- CICS SPI commands detected (continued)
 - TRACEDEST 233
 - TRACEFLAG 234
 - TRACETYPE 234
 - TRANCLASS 234
 - TRANDUMPCODE 234
 - TRANSACTION 234
 - TSMODEL 234
 - TSQNAME 234
 - TSQUEUE 235
 - UOW 235
 - UOWLINK 235
 - URIMAP 235
 - VTAM 235
 - WEB 235
 - WEBSERVICE 235
 - WORKREQUEST 235
- CICS SPI commands detectedRESETTIME
 - DUMP 231
 - ENDAFFINITY 231
- CICS variables 54
- CICSplex System Manager (CPSM)
 - API commands detected by the
 - Collector 238
- CINB request queue manager
 - diagnostics 385
- CINB transaction 20
- CINT transaction 85
- CINT user administration 87
- CIU_SCAN_DETAIL, base table 275
- CIU_SCAN_SUMMARY, base table 275
- CIU_AFF_CMD_DATA, table 271
- CIU_AFF_GRP_DATA, table 269
- CIU_AFF_INDEX_DATA, database
 - table 271
- CIU_APPLS_DESC table, database
 - structure 258
- CIU_APPLS_RESOURCES table, database
 - structure 258
- CIU_CICS_CHAINP table, database
 - structure 258
- CIU_CICS_CHAINP_T table, database
 - structure 259
- CIU_CICS_CONNP table, database
 - structure 259
- CIU_CICS_DATA table 83, 251
- CIU_CICS_DATA, table of dependencies
 - on CICS resources 251
- CIU_CONNECTIONS table 281
- CIU_CSECT_INFO, database table of
 - dependencies on CICS resources 278
- CIU_DB2_DATA table 251
- CIU_DB2_DATA, table of dependencies
 - on CICS resources 253
- CIU_EVENT_DETAIL
 - table 294
- CIU_EVENT_DETAIL table 294
- CIU_EXIT_INFO resource table 293
- CIU_EXIT_INFO table 293
- CIU_FILE_DETAIL table 282
- CIU_IMS_DATA table 251
- CIU_IMS_DATA, table of dependencies
 - on CICS resources 254
- CIU_MQ_DATA table 251
- CIU_MQ_DATA, table of dependencies
 - on CICS resources 255
- CIU_Natural_DATA table 251
- CIU_NATURAL_DATA, table of
 - dependencies on CICS resources 256
- CIU_PROGRAM_DETAIL table 284
- CIU_PROGRAM_INFO, database table of
 - dependencies on CICS resources 277
- CIU_REGION_INFO, base table 280
- CIU_RESOURCE table 296
- CIU_SQL_DATA table 251
- CIU_TDQUEUE_DETAIL table 289
- CIU_THREADSAFE_CMD, file table 259
- CIU_TRANSID_DETAIL table 286
- CIU_TRANSLATORS, database table of
 - dependencies on CICS resources 278
- CIU_TRUEEXIT_INFO table 294
- CIU_TSQUEUE_DETAIL table 291
- CIU_VERSION 297
- CIU_WEBSERV_DETAIL, file table 292
- CIU-CHAIN table 251
- CIU-CHAINP table 251
- CIU000 panel, Collector Main
 - Administration Menu
 - example 86
- CIU100 screen, Collector Operations
 - Menu
 - example 95
- CIU150 panel, Collector Statistics Menu
 - example 103, 211
- CIU151 panel, Collection Statistics Menu
 - example 104
- CIU200 panel, Collector Region
 - Configuration Menu
 - example 106
- CIU240 panel, Collector CICS Resources
 - Options
 - example 113
- CIU245 panel, Collector CICS Resources
 - Options
 - example 114
- CIU250 panel, Collector
 - DB2/MQ/IMS/CPSM Resource
 - Options
 - example 117
- CIU260 screen, Collector General Options
 - example 109
- CIU270 panel, Collector Affinities
 - Options
 - example 119
- CIU280 screen, Collector Timer Options
 - example 121
- CIU290 panel, Collector Resource Options
 - example 107
- CIU29N screen, Collector Natural
 - Options
 - example 122
- CIU300 panel, Collector Global Options
 - Menu
 - example 123
- CIU400 panel, User Administration Menu
 - example 88
 - User Administration panel 88
- CIU410 panel, Add User Menu
 - Add User panel 89
 - example 89
- CIU420 panel, Copy User Menu
 - Copy User Menu panel 91
 - example 91

- CIU440 panel, User Details Menu
 - example 93
 - User Details Menu panel 93
- CIUA01 panel, Command Flow Options
 - Command Flow Options panel 126
 - example 126
- CIUA02 panel, Command Flow ApplID list
 - Command Flow ApplID list
 - panel 129
 - example 129
- CIUA03 panel, CICS IA Command Flow Statistics
 - Command Flow Statistics panel 132
 - example 132
- CIUA04 panel, List of connected CICS regions
 - example 132
 - List of available CICS regions,
 - panel 132
- CIUA0A panel, Command Flow Application Data Collection screen
 - Command Flow Application Data Collection 130
 - example 130
- CIUAFF1/2/3 – jcl CIUJCLCA - dependency files
 - space considerations 392
- CIUAFFBL job, changes to 197
- CIUAFFRD job, to run the Affinities
 - Reporter against the Affinity database objects 170
- CIUAFFRP job, to run the Affinities
 - Reporter against the VSAM affinity data files 170
- CIUAPPL - dependency files
 - space considerations 393
- CIUCIC1/CIUCICSX - CICS tables and index
 - DB2 space allocation 394
- CIUDB2 - DB2 tables and index
 - DB2 space allocation 395
- CIUIMS - IMS tables and index
 - DB2 space allocation 396
- CIUINT1/2/3/4/5 – jcl CIUJCLCA - dependency files
 - space considerations 392
- CIUJCLCS job
 - CSECT scanner 192
- CIUJCLLS job, used to run the Load Module Scanner 184
- CIUJCLPL, running the sample batch job 77
- CIUJCLRP, Reporter job to list dependencies on resources 161
- CIUJCLTD job, used to run the Load Module Scanner 189
- CIUJCLTS job, used to run the Load Module Scanner 185
- CIUJCLXP, running the sample batch job 74
- CIUJSAMP job, Tailoring the job for your environment 207
- CIUJSAMP, running the job 207
- CIUMQ1 - MQ tables and index
 - DB2 space allocation 395

- CIUNAT - Natural tables and an index
 - Natural space allocation 396
- CIUREXIT - exit resource tables and index
 - DB2 space allocation 397
- CIURFILE - file resource tables and index
 - DB2 space allocation 397
- CIURPROG - program resource tables and index
 - DB2 space allocation 398
- CIURTDQ - transient data queue resource tables and index
 - DB2 space allocation 399
- CIURTRAN - transaction resource tables and index
 - DB2 space allocation 398
- CIURTSQ - temporary storage queue resource tables and index
 - DB2 space allocation 399
- CIURWEB - Web services resource tables and indexes
 - DB2 space allocation 400
- CIUSPAFF 411
- CIUSPAP1 423
- CIUSPAPP 407
- CIUSPDEP 436
- CIUSPDPG 416
- CIUSPEP2 433
- CIUSPEP3 434
- CIUSPEPS 431
- CIUSPPUR 425
- CIUSPTSR 418
- CIUUDBCS job
 - CSECT scanner 192
- classifying a CICS IA problem 210
- clean up job 148
- clean up utility 147, 148
- client/server interface 24
- COBOL calls
 - collecting dynamic 443
- COBOL, example, Load Module Scanner 247
- collecting data
 - affinity data 20
 - dependency data 20
 - how to 20
- collecting data with CICS IA Explorer plug-in 134
- collecting dynamic COBOL calls 443
- collection_id
 - identifying data DB2 140
- Collector
 - Affinities Options panel 119
 - changing options 105
 - global options 123
 - natural options 122
 - region-specific options 105, 107
 - Changing the options of the collection of data 97
 - CICS Resources Options panel 113, 114
 - CINB request queue manager
 - diagnostics 385
 - CINB transaction 20
 - Collection Statistics Menu panel 104
 - control record VSAM file 22
 - controlling 19

- Collector (*continued*)
 - controlling the collection of data 94
 - DB2/MQ/IMS/CPSM Resource Options panel 117
 - dependency data VSAM files 21
 - displaying the main menu screen 86
 - displaying the state of the Collector 102, 104
 - errors 209
 - General Options screen 109
 - general, region-specific, options 109
 - Global Options Menu panel 123
 - how affinity data is collected 20
 - how dependency data is collected 20
 - Main Administration Menu panel 86
 - Natural Options screen 122
 - Operations Menu screen 95
 - pausing data collection 99
 - Region Configuration Menu panel 106
 - Resource Options panel 107
 - resuming data collection 100
 - running 85
 - saving data 20
 - specifying affinity-related CICS commands
 - affinity-related CICS commands to be monitored 118
 - specifying dependency related commands to be monitored
 - CPSM, DB2, IMS, and MQ commands to be monitored 117
 - specifying region-specific options 113, 115
 - specifying region-specific options:
 - general 108
 - specifying region-specific options:
 - timers 120
 - starting data collection 95
 - Statistics Menu panel 103, 211
 - stopping data collection 101
 - table manager diagnostics 383
 - Timer Options screen 121
 - what is monitored 17
 - what is not monitored 17
- collector component
 - overview 15
- collector main administration menu panel, displaying 86
- combined affinity transaction group definitions
 - Builder 201
- combining basic affinity transaction-groups 202
- command detected
 - Built-in functionscommand 218
- command exclude list 76
- command exclude list, creating a 76
- command flow
 - overview 14
- command flow components 14
- command flow database objects
 - updating 137
- command flow functions 14
- command flow statistics panel, displaying 131
- command flow user exit 438

- commands
 - CICS API commands 219
- commands detected
 - affinity-related 12, 239
 - atomservice commands 225
 - authentication commands 217
 - bundle support commands 226
 - CICS API 12, 217
 - CICS API commands supported by
 - CICS IA 239
 - CICS FEPI API 236
 - CICS FEPI SPI 236
 - CICS SPI 12, 225
 - console support commands 219
 - CPSM 238
 - CSD commands 226
 - DB2 237
 - dependency-related 6, 217
 - distributed transaction processing (DTP) commands 219
 - event commands 229
 - events command 220
 - exception support 220
 - file control commands 220
 - IMS 237, 238
 - interval control commands 221
 - journal control commands 221
 - JVM server commands 230
 - MQ 237
 - MQ commands 231
 - named counter server
 - commands 221
 - Natural commands 239
 - non-CICS 237
 - other commands 225
 - program control commands 222
 - security commands 223
 - syncpoint commands 223
 - task control commands 223
 - temporary storage commands 223
 - trace storage commands 223
 - transient data commands 223
 - WEB commands 224, 225
 - web services addressing
 - commands 224
 - XML parser commands 225, 236
- Commands in CICS IA V3.2
 - upgrading xxix
- Commands in CICS IA V5.1
 - upgrading xxiv
- Commands in CICS IA V5.2
 - upgrading xx
- Commands in CICS IA V5.3
 - upgrading xvi
- components
 - affinity-related 10
 - command flow 14
 - dependency-related 4
- compressing affinity data 176
- configuration
 - CICS environment 31
 - collection 46
 - DB2 35
 - full configuration 35
 - short configuration 35
- configuring
 - CICS considerations 46

- configuring (*continued*)
 - CICS IA controlling region 47
 - collecting DB2 commands 48
 - collection environment 50
 - command flow collector 49
 - connect to CICS IA plug-in 45
 - controlling with CICS IA plug-in 48
 - create collection environment 59
 - create new DB2 environment 43
 - create upgraded collection
 - environment 71
 - creating upgraded DB2
 - environment 65
 - existing DB2 environment 64
 - existing environment 69, 155
 - load IVP data 45
 - modify settings 34
 - new DB2 environment 38
 - run configuration exec 33
 - share VSAM files 48
 - upgrade collection configuration 69
 - upgrading DB2 configuration 63
- configuring CICS IA
 - setup 31
- configuring CICS IA Explorer
 - plug-in 134
- connections table 281
 - CICS resource objects 281
 - structure 281
- contacting IBM support 319
- contacting IBM Support 214
- control record VSAM file 22
- controlling the collection of dependency
 - and affinity data 94
- copy user menu 90
- correlating Load Module Scanner and Reporter output 247
- CPSM
 - commands detected by the
 - Collector 238
 - monitoring dependency-related
 - commands 117
- creating a command exclude list 76
- creating a program exclude list 74
- creating a resource compression list 77
- creating a transaction exclude list 75
- creating a translation table 83
- creating your own program exclude,
 - transaction exclude, and resource
 - compression lists 74
- CSECT Load Module Scanner
 - example report 193
 - how to run 191
- CSECT Module Scanner tables
 - affinity tables and indexes 402
- CSECT scanner
 - base tables 277
 - CIUJCLCS job 192
 - CIUUDBCS job 192
 - object views 279
 - overview 27
 - printed report 193
- CSECT Scanner objects
 - CIU_CSECT_INFO table 278
 - CIU_PROGRAM_INFO table 277
 - CIU_TRANSLATORS table 278
 - structure 277

- CSECT Scanner objects (*continued*)
 - V_CIU_CICS_LINKED view 279
 - V_CIU_CSECT_TRANS view 279
- customizing command flow data
 - collection 125
- Customizing the CICS IA Natural Interface 82

D

- data sets processed report, Builder 203
- data space allocation
 - space considerations 388
- database objects, Affinity
 - updating 137
- database objects, command flow
 - updating 137
- database objects, Dependency
 - DB2 35
 - structure 251
 - updating 135
- database structure 251
 - application dependency views 261, 264, 266
 - CIU_APPLS_DESC table 258
 - CIU_APPLS_RESOURCES table 258
 - CIU_CICS_CHAINP table 258
 - CIU_CICS_CHAINP_T table 259
 - CIU_CICS_CONNP table 259
 - dependency views 261
 - V_CIU_CICS_INDS2 261
 - V_CIU_DB2_INDS2 262
 - V_CIU_DB2_RES 261
 - V_CIU_DB2_TRANSID_DET 265
 - V_CIU_DB2_WEBSERV 267
 - V_CIU_IMS_INDS2 263
 - V_CIU_IMS_TRANSID_DET 266
 - V_CIU_IMS_WEBSERV 268
 - V_CIU_MQ_INDS2 263
 - V_CIU_MQ_TRANSID_DET 265
 - V_CIU_MQ_WEBSERV 268
 - V_CIU_TRANSID_DET 264
 - V_CIU_WEBSERV 266
- date formatter diagnostics 385
- DB2
 - configuration 35
 - monitoring dependency-related
 - commands 117
- DB2 commands detected by the
 - Collector 237
- DB2 environment
 - creating upgraded 65
- DB2 objects
 - CIU_THREADSAFE_CMD table 259
 - CIU_WEBSERV_DETAIL table 292
- DB2 security 405
- DB2 space allocation
 - affinity tables and indexes 400
 - CICS tables and index -
 - CIUCIC1/CIUCICSX 394
 - DB2 tables and index - CIUDB2 395
 - exit resource tables and index -
 - CIUREXIT 397
 - file resource tables and index -
 - CIURFILE 397
 - IMS tables and index - CIUIMS 396
 - MQ tables and index - CIUMQ1 395

- DB2 space allocation (*continued*)
 - program resource tables and index - CIURPROG 398
 - space considerations 393
 - temporary storage queue resource tables and index - CIURTSQ 399
 - transaction resource tables and index - CIURTRAN 398
 - transient data queue resource tables and index - CIURTDQ 399
 - Web services resource tables and indexes - CIURWEB 400
- DB2 Threadsafe table 259
 - structure 259
- DB2 variables 39
- DB2 variables for CINT 58
- DB2, when creating the Dependency database objects 35
- Deleting by Application ID platform 145
- Deleting by Collection ID 144
- Deleting old versions of programs platform 146
- Deleting TSQ or ENQ 145
- dependencies and affinities collected, details of 217
- dependency base tables 251
- Dependency base tables
 - CIU_CICS_DATA table 251
- dependency data VSAM files 21
- dependency database objects 23
- Dependency database objects
 - DB2 35
 - structure 251
 - updating 135
- Dependency database objects
 - structure 251
- dependency facilitating tables
 - structure of the database 258
- Dependency objects
 - CIU_DB2_DATA table 253
 - CIU_IMS_DATA table 254
 - CIU_MQ_DATA table 255
 - CIU_NATURAL_DATA table 256
- Dependency Reporter 161
 - overview 25
- dependency reporter output 247
- dependency views
 - database structure 261, 264, 266
- dependency-related commands detected by the Collector 217
- dependency-related components 4
- dependency-related EXEC CICS commands 6
- dependency-related functions 3
- detailed report (Load Module Scanner)
 - creating 186, 189
 - output contents 187
 - output example 187
- details of dependencies and affinities collected 217
- diagnostics
 - CINB request queue manager 385
 - data formatter 385
 - table manager 383
- displaying Collector statistics for a specified region 102, 104

- displaying the add user menu 89
- displaying the cics ia command flow options 125
- displaying the cics ia user administration 87
- displaying the copy user menu 90
- DLI commands detected by the Collector 237
- document services commands
 - CICS API commands 224
 - presentation commands 224
- dump facility 211

E

- eb service resource table 292
- Eclipse
 - client front end to CICS IA 24
 - client, front end to CICS IA 24
 - graphical user interface (GUI) 24
- ENQ 229
- errors
 - abend codes 319
 - Collector 209
 - messages 319
- example, CICS report 164
- examples, Load Module Scanner 247
- EXITPROGRAM 229
- Explorer, solving problems
 - obtaining an error log 213
 - obtaining configuration details 214
- External Interface 407, 411, 416, 418, 423, 425, 430, 431, 433, 434, 436

F

- facility, affinity lifetimes 9
- file resource table 282
 - CICS resource objects 282
 - structure 282
- Front End Programming Interface (FEPI)
 - API commands detected by the Collector 236
 - SPI commands detected by the Collector 236
- function code values
 - collector CINB 385
 - collector table 383
- functions
 - command flow 14
 - dependency-related 3

G

- general variables 42, 57
- getting started with CICS IA 29
- global, affinity relations 8
- granting access to the plans and tables 36
- grouping programs 80
- grouping transactions 80
- Grouping transactions and programs 80

H

- HEADER statements, Builder 200
- hover help
 - CICS IA plug-in 445
- how to use this information xiv

I

- IBM Support
 - information data sheet 215
 - when to contact 214
 - working with 215
- IBM support, contacting 319
- Identifying data to delete 141
- IMS
 - monitoring dependency-related commands 117
- IMS commands detected by the Collector 237, 238
- information data sheet, for use with IBM Support 215
- INQUIRE commands 229
- INQUIRE JVMPROFILE 229
- INQUIRE MVSTCB 229
- INQUIRE OSGIBUNDLE 229
- INQUIRE OSGISERVICE 229
- INQUIRE REQID 229
- INQUIRE RRMS 229
- INQUIRE STORAGE 229
- INQUIRE STREAMNAME 229
- INQUIRE SUBPOOL 229
- INQUIRE TSPPOOL 229
- INQUIRE UOWDSNFAIL 229
- installation
 - Natural support 81
- installation verification program (IVP)
 - how to run 62, 73
- inter-transaction affinity 8
- Introduction 1
- IPFACILITY 229

J

- jcl CIUJCLCC control file
 - space considerations 391

L

- Link3270, affinity relations 8
- list of available CICS regions, displaying 132
- list of connected CICS regions 132
- Load Module Scanner
 - affinity tables and indexes 401
 - assembler language example 247
 - base tables 275
 - CIUJCLLS job, modifying 184
 - CIUJCLTD job, modifying 189
 - CIUJCLTS job, modifying 185
 - COBOL example 247
 - creating a detailed report 186, 189
 - creating a summary report 184, 185
 - detailed report (example) 187
 - output 247
 - overview 26

- Load Module Scanner (*continued*)
 - specifying run time values 184, 185, 189
 - summary report (example) 185
- Load Module Scanner and Reporter, correlating output 247
- Load Module Scanner objects
 - CIU_SCAN_DETAIL table 275
 - CIU_SCAN_SUMMARY table 275
 - structure 275
- logon, affinity lifetimes 9
- LUName, affinity relations 8

M

- main administration menu panel, displaying the Collector 86
- Managing affinity data 147
- managing CICS IA data 139
- Managing collected data 140
- Managing the data that you collect 139
- messages 319
- Messages that CICS IA can issue 319
- Migrating application definitions
 - migrating from version 3.1 80
- migrating from version 3.1
 - migrating application definitions 80
- modifying affinity transaction groups 175
- modifying clean up utility job 148
- Modifying the Exclusive Work Area (EWA) size 82
- Modifying the Natural Name List size. 82
- MQ
 - monitoring dependency-related commands 117
- MQ commands detected by the Collector 237
- MVS link list modules 83

N

- Natural 82
- Natural space allocation
 - Natural tables and an index - CIUNAT 396
- Natural support
 - preparing to use CICS IA 81
- Natural support installation 81
- new commands xvi, xx, xxiv, xxix
- non-CICS commands detected by the Collector 237

O

- objects, Affinity
 - structure 269
- objects, CICS regions
 - structure 280
- objects, CSECT Scanner
 - structure 277
- objects, Load Module Scanner
 - structure 275
- obtaining an error log
 - solving problems, Explorer 213

- obtaining configuration details
 - solving problems, Explorer 214
- output data set variables 39, 51
- output of the CSECT Load Module Scanner (example) 193
- output of the Dependency Reporter (example) 164, 167, 168
- output of the Load Module Scanner (example) 185
- overview
 - command flow feature 14
 - problem determination 209
- Overview 1

P

- pausing data collection 99
- PERFORM SHUTDOWN 231
- permanent, affinity lifetimes 9
- PLITDLI commands detected by the Collector 238
- plug-in
 - graphical user interface (GUI) 4, 14
- plug-in Level
 - solving problems 213
- plug-in resource table
 - structure of CICS IA database 296
- plug-in Resource table: CIURESTB/X
 - DB2 space allocation 403
- plug-in, solving problems
 - viewing Eclipse plug-ins 214
- preparing to use CICS IA
 - Natural support 81
- prerequisites 2
- Presentation commands
 - BMS commands 218
- problem determination
 - abend codes 319
 - classifying the problem 210
 - contacting IBM Support 214
 - dealing with errors 210
 - dump facility 211
 - information data sheet 215
 - messages 319
 - overview 209
 - preliminary checks 210
 - trace facility 211
- problem solving 209
- process, affinity lifetimes 9
- program exclude list, creating a 74
- program exclude, transaction exclude, and resource compression lists, creating your own 74
- program table 284
 - structure 284
- pseudoconversation, affinity lifetimes 9

R

- RACF 405
- reason code values
 - collector CINB 385
 - collector table 384
 - date formatter 385
- removing resources 147

- Reporter
 - Affinities Reporter 25
 - CICS report (example) 164
 - CIUJCLRP job, modifying 161
 - DB2 report (example) 167
 - Dependency Reporter 25
 - how to run 161
 - IMS report (example) 168
 - MQ report (example) 168
 - overview 24
 - specifying run time values 161
 - threadsafe Reporter 25
- reports
 - affinity report, Affinities Reporter 171
 - Builder error 205
 - data sets processed 203
 - dependency, affinities, threadsafe 24
 - empty transaction groups 204
 - group merge 204
- requirements 2
- resource compression list, creating a 77
- resource names
 - identifying patterns 139
- resuming data collection 100
- running clean up utility 148
- running the Affinities Reporter 169
- running the Builder 197
- running the CIUJSAMP job 207
- running the Collector 85
- running the Collector for the first time 85
- running the CSECT Load Module Scanner 191
- running the Dependency Reporter 161
- running the IVP (installation verification program) 62, 73
- running the Load Module Scanner
 - Load Module Scanner
 - how to run 183
- running the report 174
- running the reporter 161, 177
- running the sample batch job
 - CIUJCLPL 77
- running the sample batch job
 - CIUJCLXP 74
- Running the sample DB2 query 207
 - running the CIUJSAMP job 207

S

- sample analysis threadsafe report
 - threadsafe 179
- saving data 20
- scanners
 - CSECT 27
 - Load Module 26
- Scanners
 - CSECT 191
 - Load Module 183
- SECURITY 231
- setup for CICS IA
 - creating the Dependency database
 - objects
 - DB2 35
 - overview 31
 - signon, affinity lifetimes 9

- solving problems 209, 213
- space allocation, space
 - considerations 387
- space considerations
 - calculating CICS data space 389
 - calculating DB2 data space 389
 - calculating IMS data space 389
 - calculating MQ data space 390
 - calculating Natural data space 390
 - calculating resource data space 390
 - Calculating the space required for
 - Interdependency Collection 388
 - calculation for the space required for
 - affinity collection 391
 - control file: CIUCNTL 391
 - data space allocation 388
 - DB2 space allocation 393
 - dependency Files – CIUAFF1/2/3 –
 - jcl CIUJCLCA 392
 - dependency Files – CIUAPPL 393
 - dependency Files – CIUINT1/2/3/4/5 –
 - jcl CIUJCLCA 392
 - required data, CICS regions 387
 - total Interdependency data space
 - calculation for
 - Interdependency 391
 - Values required for each CICS
 - region 387
 - VSAM data set allocation 391
- space considerations, space
 - allocation 387
- SPUFI 207
- SSL REBUILD 231
- starting and stopping
 - PLT 82
- starting and stopping from the PLT 82
- starting data collection 95
- stopping data collection 101
- Stored Procedure 407, 411, 418, 430
- Stored Procedures 407, 416, 423, 425,
 - 431, 433, 434, 436
- Stored Procedures for Application
 - Deployment 430
- structure of the CICS regions objects 279
- structure of the database 251
- structure of the DB2 objects 281
- summary of changes xix
- summary report (Load Module Scanner)
 - creating 184, 185
 - output contents 184
 - output example 185
- supplied modules required in the MVS
 - link list 83
- system, affinity lifetimes 9

T

- table
 - CIU_EVENT_DETAIL 294
- table identifier values
 - collector table 383
- Tailoring the CIUJSAMP job for your
 - environment 207
- taking a dump 211
- TCP/IP services commands
 - CICS API commands 224
 - presentation commands 224

- temporary storage compression 176
- temporary storage queue resource
 - table 291
 - CICS resource objects 291
 - structure 291
- terminal control and information
 - commands
 - CICS API commands 218
 - presentation commands 218
- the dependency reporter
 - output 164
- threadsafe Reporter
 - overview 25
- Threadsafe table, DB2
 - structure 259
- total Interdependency data space
 - calculation for Interdependency
 - space considerations 391
- trace facility 211
- tracing 211
- transaction affinities 7
 - what are they? 7
- transaction affinities lifetimes, worsening
 - of 10
- transaction exclude list, creating a 75
- transaction group definitions,
 - producing 173
- transaction resource table 286
 - structure 286
- transaction security 405
 - DB2 security 405
 - RACF categories for transactions 405
- transaction-system affinity 8
- transient data queue resource table 289
 - CICS resource objects 289
 - structure 289
- translation table 83
- type and function mapping for monitored
 - commands
 - CICS input parameters 302

U

- UDB database
 - CICS IA 151
 - create 155
 - loading IVP 157
 - update 157
- updating the Affinity database
 - objects 137
- updating the command flow database
 - objects 137
- updating the Dependency and Affinity
 - database objects 135
- updating the Dependency database
 - objects 135
- upgrade 32
- user administration 87
- user command flow options 129, 130
- user details menu 92
- user ID, affinity relations 8

V

- V_CIU_AFFINITY, affinity view 272
- V_CIU_CICS_INDS2, database
 - structure 261
- V_CIU_CICS_LINKED, view 279
- V_CIU_CICS_TRANSID_DETAIL,
 - database structure 264
- V_CIU_CICS_WEBSERV, database
 - structure 266
- V_CIU_CSECT_TRANS, view 279
- V_CIU_DB2_INDS2, database
 - structure 262
- V_CIU_DB2_RES, database
 - structure 261
- V_CIU_DB2_TRANSID_DET, database
 - structure 265
- V_CIU_DB2_WEBSERV, database
 - structure 267
- V_CIU_IMS_INDS2, database
 - structure 263
- V_CIU_IMS_TRANSID_DET, database
 - structure 266
- V_CIU_IMS_WEBSERV, database
 - structure 268
- V_CIU_MQ_INDS2, database
 - structure 263
- V_CIU_MQ_TRANSID_DET, database
 - structure 265
- V_CIU_MQ_WEBSERV, database
 - structure 268
- V_CIU_SCAN_TRDSAFE table 276
- V_CIU_TRUEEXIT_INFO table 294
- version table
 - structure of CICS IA database 297
- viewing details of a command flow
 - collector user 92
- viewing Eclipse plug-ins
 - solving problems, plug-in 214
- VSAM data set allocation
 - space considerations 391

W

- W 292
- Web service resource table
 - CICS resources objects 292
 - CICS resources structure 292
- web services 134
- what this information is about xiii
- who this information is for xiii
- worsening of transaction affinities
 - lifetimes 10

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

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

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

For license inquiries regarding double-byte character set (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.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan*

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 jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

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

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, should contact:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

Such information may 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 Program License Agreement or any equivalent agreement between us.

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

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.



SC34-7456-00

