C\Prof:
Transaction Profiling for CICS®

Ezriel Gross - Circle Software
ezriel@circle-us.com
March, 2017

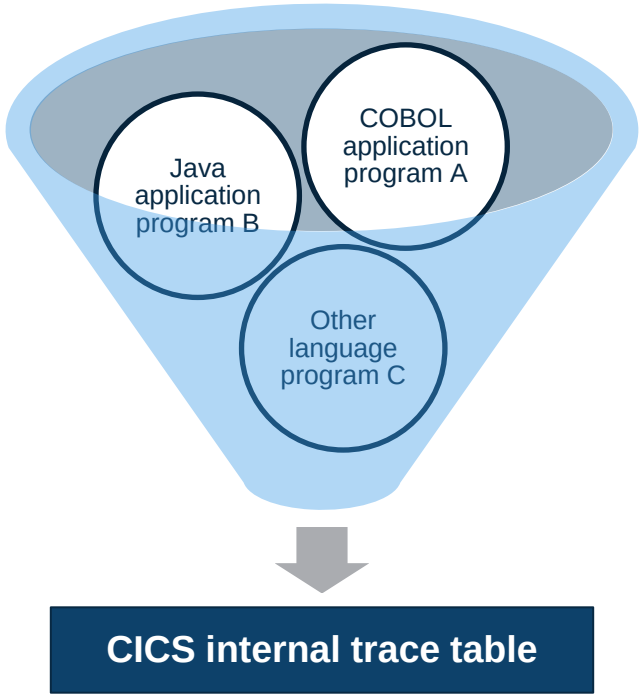
Agenda

- **The CICS diagnostic ecosystem**
  - The CICS internal trace
  - The problem with monitors and exits
- **Introducing C\Prof**
- **How does it work?**
- **Minimal impact on CICS**
- **Highlights**
  - Application view of the trace
  - Three modes: profiler, record, and snap
  - Development vs. production
- **Who can benefit?**
- **What do I need?**
- **Quick start**
- **Profiler**
  - Transaction List (and filter)
  - Application Events (line action S)
    - Latencies and response codes
    - Java and JCICS
  - Application Commands (line action C)
  - Program Analysis (line action P)
  - Transaction Overview (line action O)
  - Trace events (deep dive analysis)
- **Auxiliary trace data sets**
  - Snap (snapshot) the past
  - Record the future
- **Getting more from the trace**
  - Adjusting trace point levels
  - Creating user trace entries
- **C\Prof case study:** Profiling transactions where Java calls legacy business Cobol applications

The CICS diagnostic ecosystem

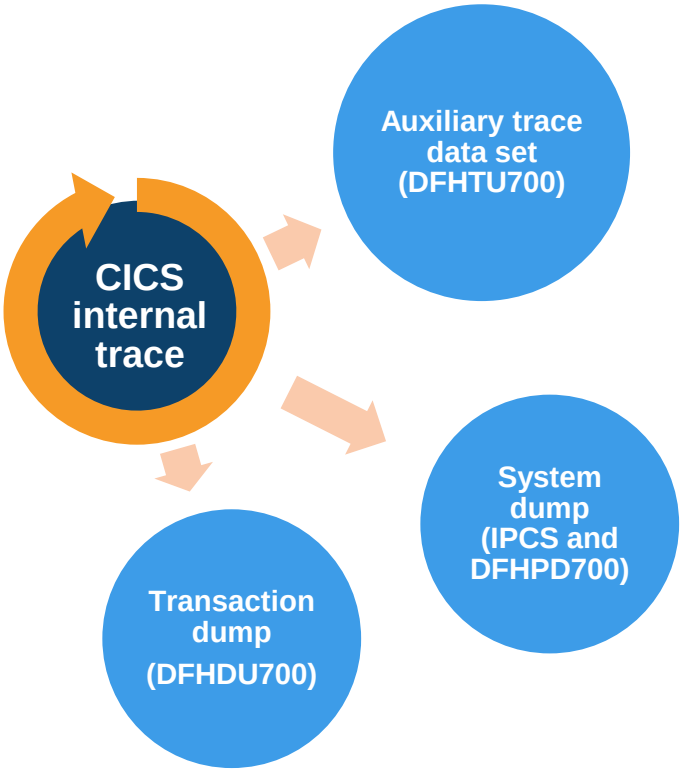
The CICS internal trace table

- MVS 64-bit (above-the-bar) storage held inside the CICS region
- Allocated during CICS initialization – you control the size (16KB to 1 GB)
- Contains *trace entries* produced by CICS – wraps around when full
(a “circular buffer”)
- All CICS regions have one – even when internal tracing not started
- Always contains *exception* entries (even if no trace destinations are started)



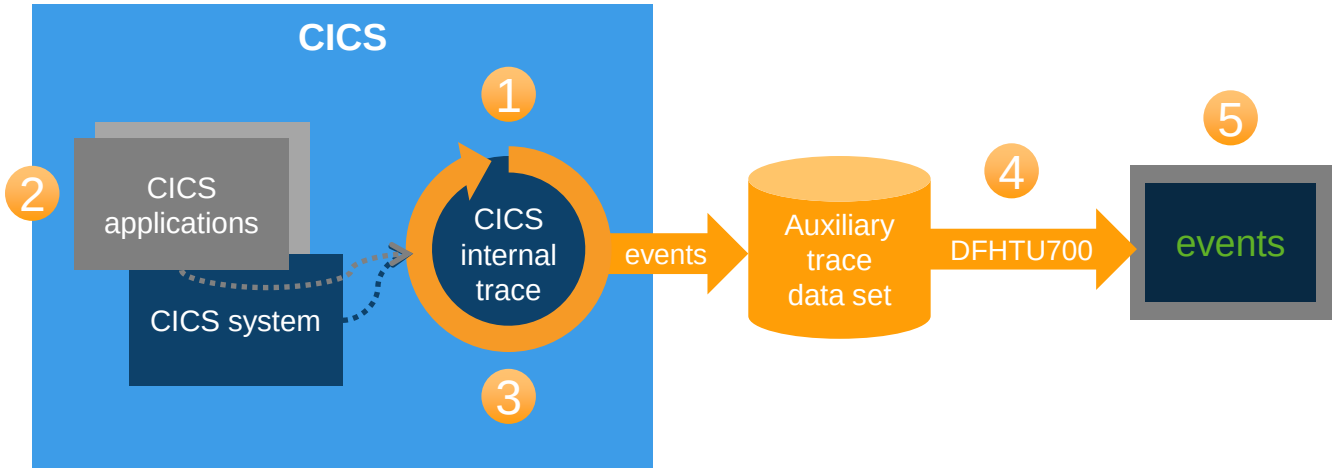
Tapping into the trace...

- The standard tools use the CICS internal trace to capture trace information
- It is a rich source of information, but the tools have their limitations…



Today: the CICS auxiliary trace

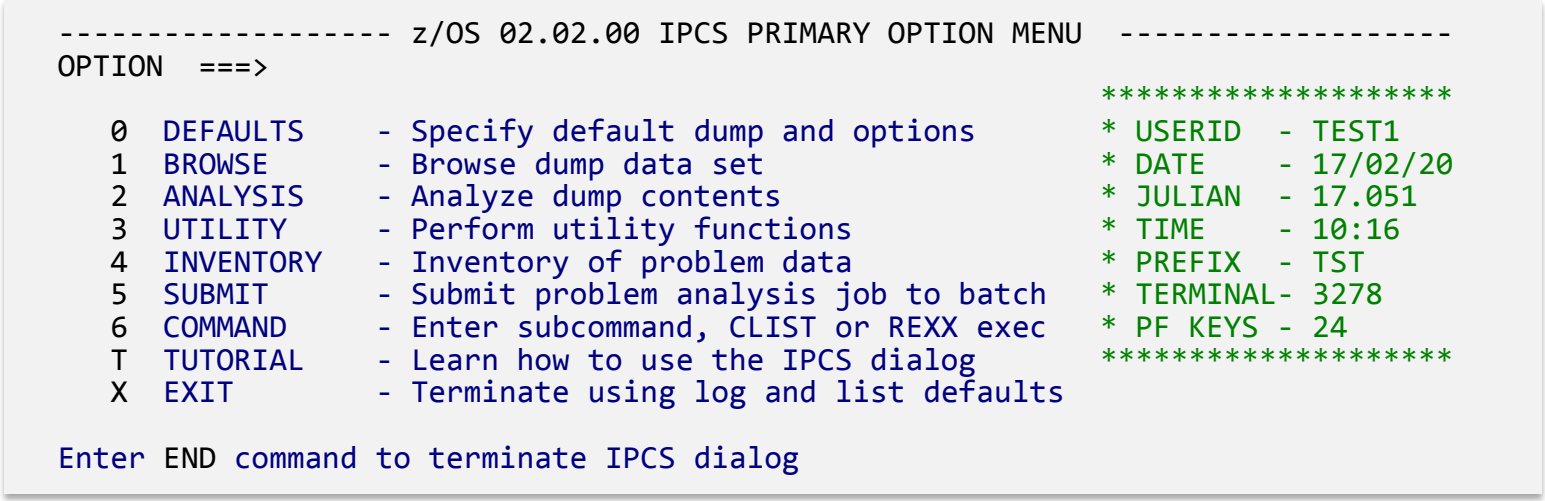
1. Start auxiliary trace
2. Recreate problem
3. Stop auxiliary trace
4. Format
5. Analyze



Objective is to complete steps 1 to 3 as quickly as possible to minimize CPU expense!

System and transaction dumps

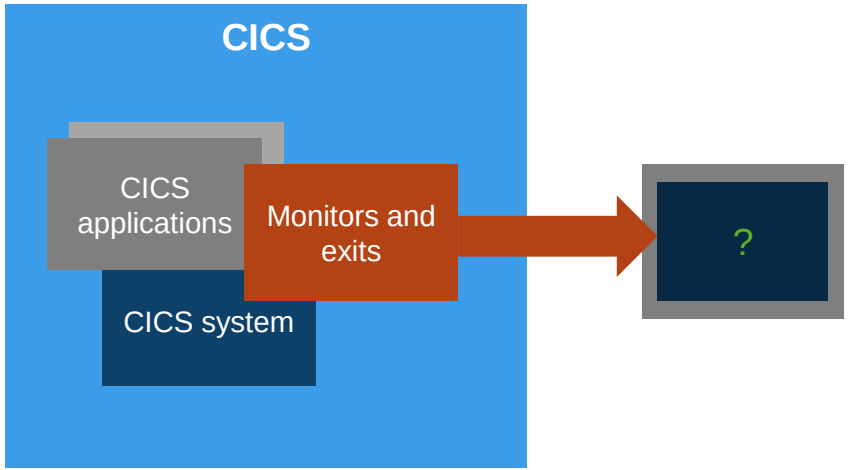
- DFHPD700 and IPCS: Format system dump
- DFHDU700: Format transaction dump
- Only useful when CICS creates a dump in the first place (ABEND)
  - Need other tools to uncover problems not caused by ABEND



Interfering with CICS: monitors and exits

Not supplied with CICS

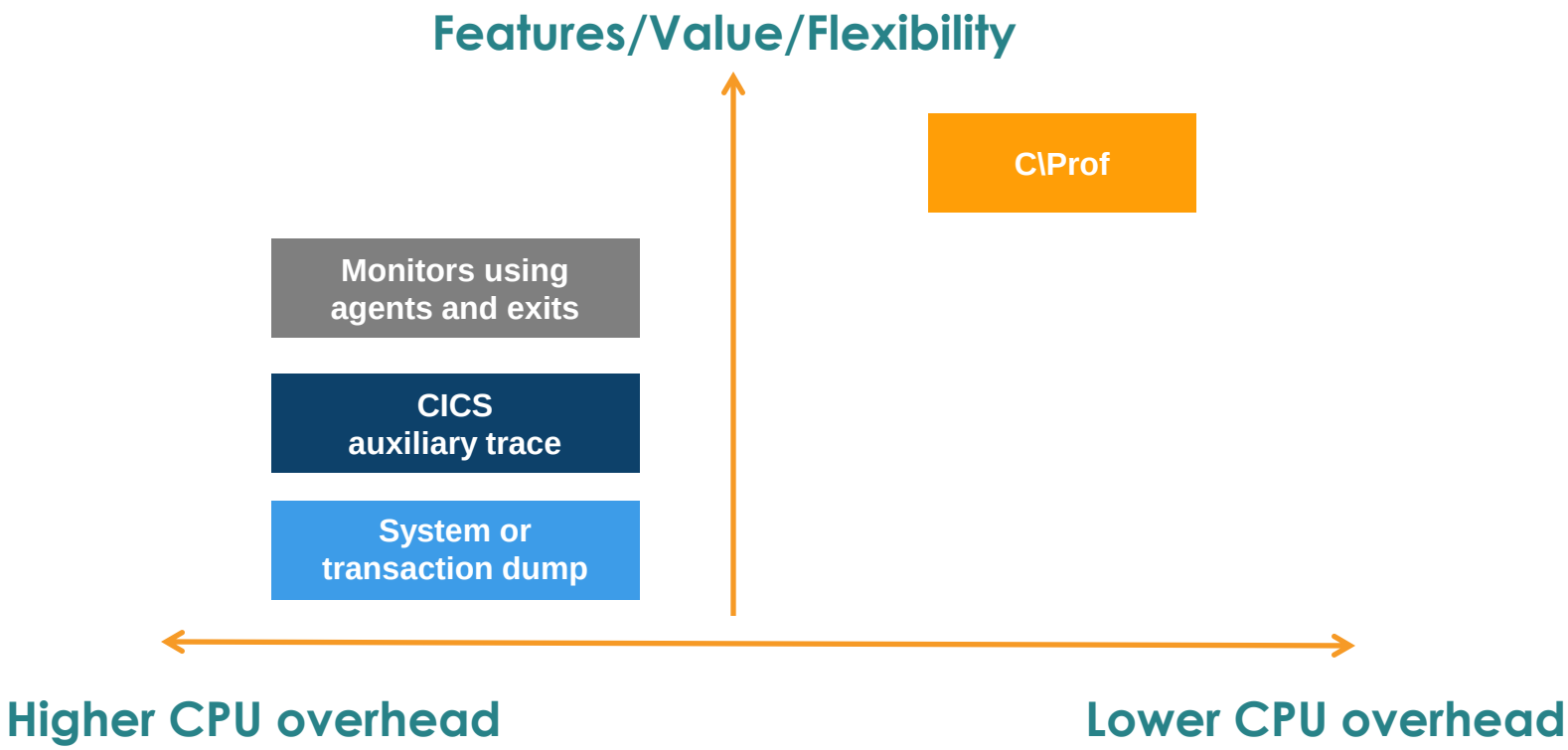
Requires permanent changes inside CICS itself to function



May require program changes

Can introduce CPU overhead

The state of CICS trace tools today



C\Prof

Introducing C\Prof

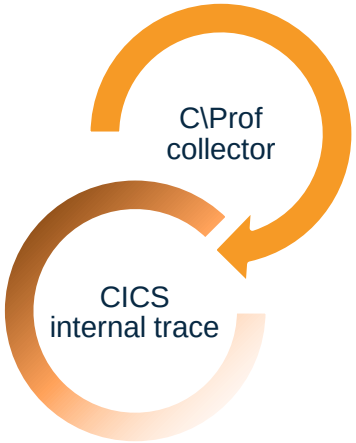
- A completely new approach to trace capture
- Uses significantly less CPU than traditional tools
- Does not require changes to CICS
- With C\Prof, the CICS trace becomes:
  - Inexpensive to capture
  - Simple to interpret
- C\Prof *unlocks* the hidden value of the trace
  - Low CPU usage means you can run it in production
  - Ideal for permanent activation in development environments

How C\Prof works...

low overhead cross-memory collection that runs from a separate MVS address space

How does it work?

- **Collector runs in separate MVS address space**
  - Peeks inside CICS to look at CICS internal trace
  - CICS is unaware that this is happening!



- **The CICS internal trace is collecting all the time**
  - C\Prof collector pulls it into archive data sets (GDG or dynamic) or to auxiliary trace data sets
  - Level of detail is only limited to what is stored in the trace
  - Collect all the time, in a short burst, or take a quick snapshot!

- **Collect one region or multiple with same collector**
  - Use C\Prof to filter and sort transactions, view application events, dive into the trace events themselves
  - Supports regions using MRO

Minimal impact on CICS

- **No additional code inside CICS**
  - No monitors
  - No exits
  - No custom modifications
  - No hassle
- **No CICS system or resource changes**
  - Just start the CICS internal trace – that's all
- **Runs in separate MVS address space**
  - The result is that C\Prof imparts *significantly less CPU overhead* than traditional trace debugging aids

C\Prof completely replaces the CICS Auxiliary Trace...

(you will never want to use it again!)

Highlights: C\Prof

- Simple to get started (< 30 minutes)
- Transaction data supplied in near-real time
- An “application” view of your trace events:
  - Collect event data from the CICS internal trace
  - Find your transaction
  - Drill down to application events, performance information, transaction breakdown by program, trace event “deep dive”
- Adjustable trace levels
  - Add additional trace points for more detail
- All the features of the CICS auxiliary trace but much more

Multiple trace capture modes

- **Snap**
  - Take a point in time snapshot of the CICS internal trace
  - Looks “backward” to see what recently occurred in CICS
  - Can capture recent events, even if the collector was not running at the time!

- **Record**
  - Record the ***future*** contents of the CICS internal trace to an auxiliary trace data set
  - Like the CICS auxiliary trace, but doesn’t involve CICS

- **Profile**
  - Capture the ***future*** contents of the CICS internal trace C\Prof archive data sets
  - Constructs an application perspective of the trace events

Use in development and in production

- Modes to suit any environment
  - Ideal for ***permanent activation*** in a low volume workload environment
  - Use ***short burst*** analysis and ***snap*** feature in production to minimize impact
- Specify a collection **time limit** to “take a sample” from your production environment
- ***“Oh! What was that? What just happened?”***
  - Use **Snap** to quickly capture recent history (you don’t even need to have the collector running when the problem occurred!)

Feature	Development	Production
Snap	✓	✓
Record (short burst)	✓	✓
Profiler (short burst)	✓	✓
Record (no limit)	✓	
Profiler (no limit)	✓	

Who benefits?

Application Developers

- **Debug** applications by viewing application events (EXEC CICS, JCICS, DB2, IMS, MQ, etc.)
- **Measure** transaction performance during development
- **Identify** delays before they present in production

Project Leads

- **Explore** code coverage and application completeness during code review
- **Compare** the before and after picture of a program to validate changes
- **Verify** that application changes have not adversely impacted response time

Analysts

- **Measure** all aspects of application performance and identify bottlenecks
- **Examine** interdependencies between programs and the resources they use

Testers

- **Validate** the performance of applications before they go into production
- **Capture** and share problems automatically with Application Developers as they occur

Systems Programmers (Production Support)

- **Diagnose** a problem in production using short burst analysis
- **Capture** auxiliary trace data sets as soon as problems occur (no need to recreate the problem)

What you need...

1. **CICS Transaction Server version 4.2, 5.1, 5.2, or 5.3**
   - A single C\Prof collector can support any mix of versions simultaneously
2. **The CICS internal trace turned ON with sufficient trace table size**
   - 32 MB is sufficient for moderately busy systems
   - The trace table resides in 64-bit storage - a slight increase should not impact CICS
3. **z10 hardware equivalent or higher**
   - Better performance gained through newer instructions
   - z9 hardware with upgrades is acceptable
4. **A small amount of additional CPU capacity**
   - Required on the LPAR where CICS is running
   - Collection needs to keep up with the speed at which CICS is writing trace entries
5. **Sufficient DASD space to save trace data for profiling**
   - Particularly if using dynamic data set allocation instead of GDG

Getting started

Quick and easy collector setup

1. Install the C\Prof libraries using the instructions supplied in the C\Prof *readme* file.
2. Start the CICS internal trace in each region you wish to monitor
3. Configure C\Prof for use in your environment using the *C\Prof User's Guide* and by completing each step outlined on the **Administer** ISPF dialog panel…

Tips:

- Invoke the C\Prof ISPF dialog quickly from ISPF option 6 command shell:

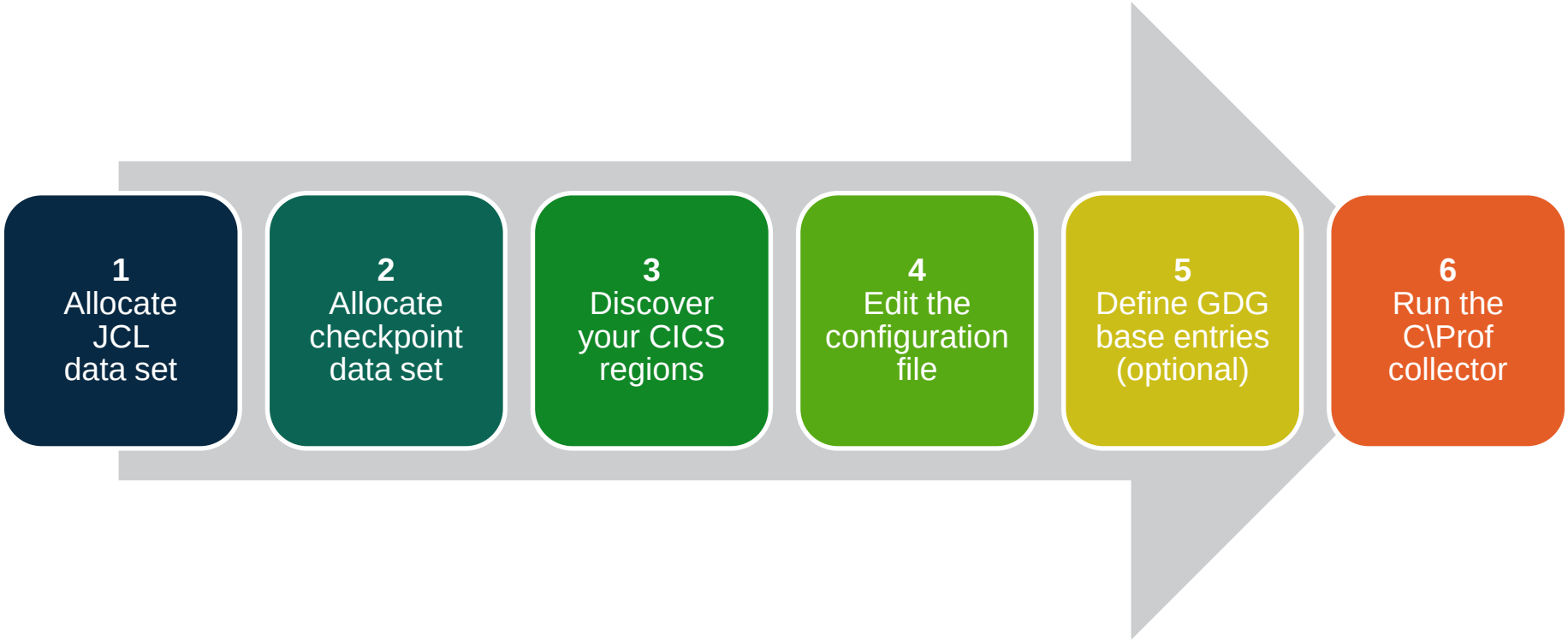
```
EX 'TXC.V1R1M0.STXCEXEC(TXCOREXX)' 'TXC.V1R1M0'
```

- Don't worry about the product registration key – ***you have a three month trial period***

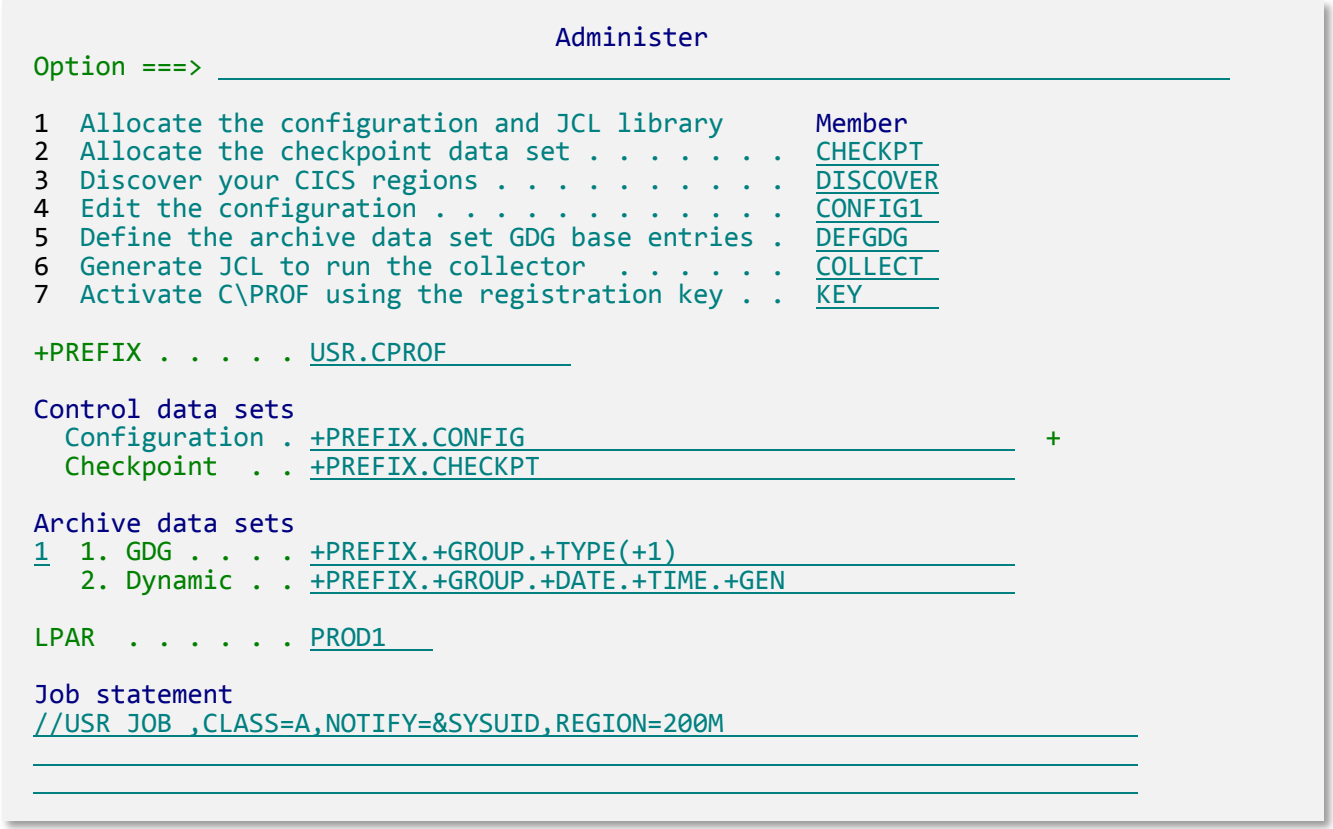
Activating the CICS internal trace…

- Activate at any time using CICS transaction `CETR`:
  1. Set Internal Trace Status to `STARTED`
  2. Set Internal Trace Table Size to minimum of `32768` (32 MB)
  3. Set Master System Trace Flag to `ON`
- Activate at CICS startup using SIT parameters:
  - `SYSTR=ON`
  - `INTTR=ON`
  - `TRTABSZ=32768`

Administration by the numbers...



Administration by the numbers...



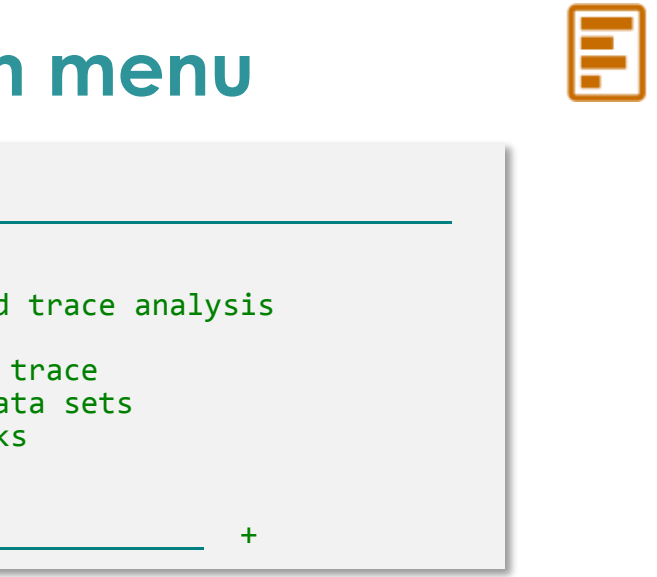
Administration by the numbers...

Start the CICS internal trace for each region that you wish to monitor - and then:

1. **Allocate the configuration and JCL library**
   - Used to store the configuration file and reusable JCL generated by the ISPF dialog
2. **Allocate the checkpoint data set**
   - VSAM KSDS used to register the archive data sets created during collection – acts as an index for the ISPF dialog
3. **Discover your CICS regions (and create a configuration file)**
   - Searches your LPAR for active CICS regions and creates a default configuration file
   - Choose between GDG archive data sets or dynamically allocated archive data sets
4. **Edit the configuration file (optional)**
   - Review collection settings and make changes as required (region grouping, data set naming, collection time limit, transaction filtering, trace levels, etc.).
5. **Define the archive data set GDG base entries (GDG only)**
6. **Generate JCL to run the collector (and then submit it!)**

Primary Menu Option 1: Profiler

C\Prof ISPF dialog: primary option menu



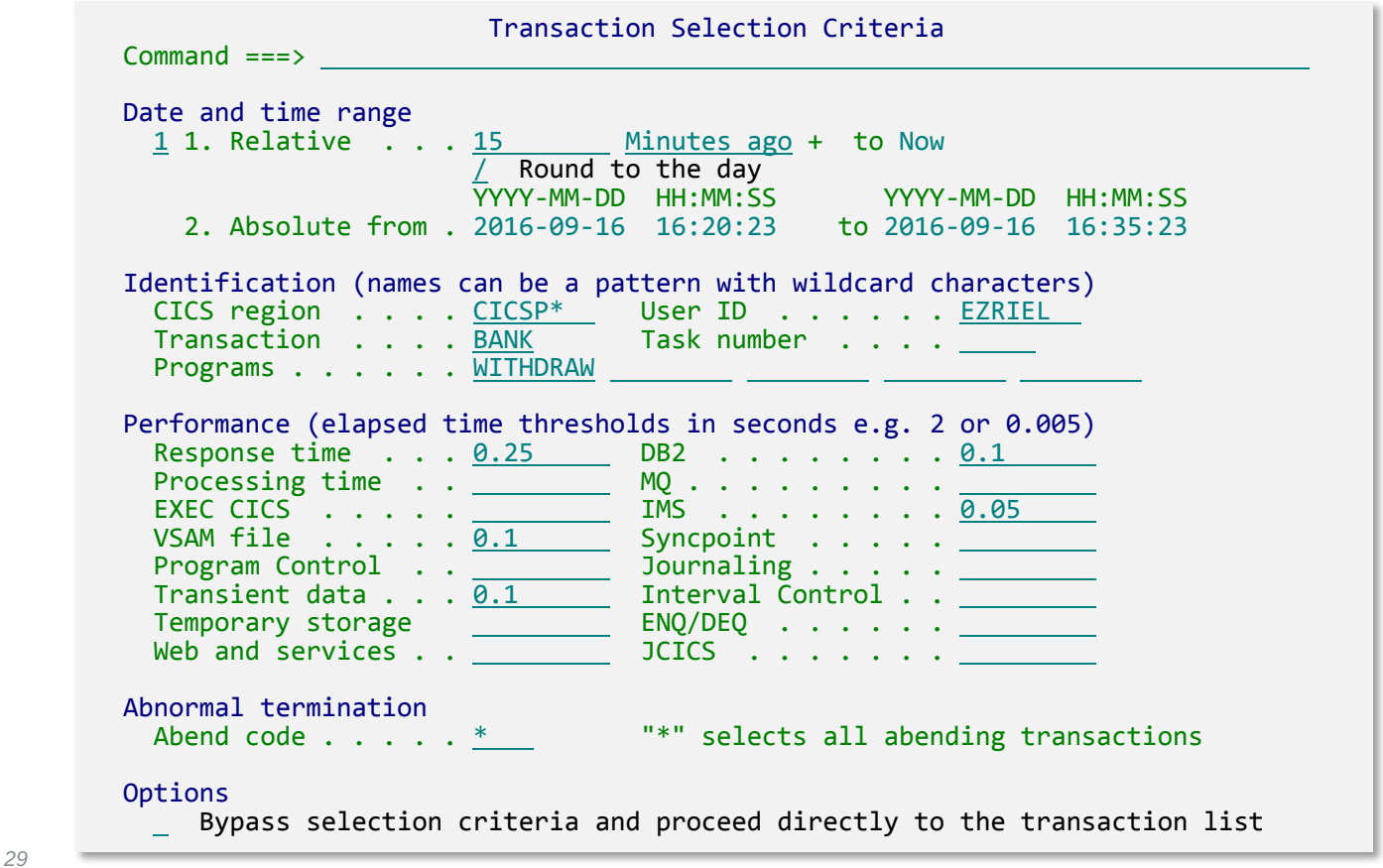
Simple to use, easy to understand:

1. **Add/select a configuration file**
2. **Select option 1 to start**

Tip: If your team uses the ***same configuration file***, they all get universal access to the same collected data.

Alternatively, you can use ***multiple collectors/configuration files*** for different CICS regions, groups, transactions, Java, or the individuals themselves…

Profiler: Transaction Selection Criteria



Selection criteria is used to retrieve only those transactions of interest to you.

Use performance thresholds to quickly identify exceptional transactions.

Specify your criteria then press **Enter** to see the **Transaction List**…

Profiler: Transaction List

File Edit Filter Form Help

C\PROF Transaction List — Line 00000001
Command ===> — Scroll ===> PAGE

/	Date	Time	Tran	Program	Second Program	Task Number	Userid	APPLID	ABEND Code	Response Time	First Dispatch Time	Application Processing Time	SYNCPOINT Time	EXEC CICS Time	VSAM File Time	Transien Dat Tim
_	2016-09-16	16:33:26.460891	CJSA		DATABUS	389	CICSUSER	DEVSYS01		0.044011	0.000000	0.039357	0.001078	0.026152	0.000096	0.00000
_	2016-09-16	16:30:27.347287	CJSA		DATABUS	386	CICSUSER	DEVSYS01		0.050968	0.000000	0.047827	0.000202	0.027953	0.000087	0.00000
_	2016-09-16	16:30:23.344503	CJSA		DATABUS	385	CICSUSER	DEVSYS01		1.297130	0.000000	1.282175	0.000255	0.123445	0.000098	0.00000
_	2016-09-16	16:29:56.090126	CJSU			384		DEVSYS01		0.157322	0.000000	0.156889	0.000085	0.000204	0.000000	0.00000
_	2016-09-16	16:29:56.089944	CJSU			383		DEVSYS01		0.233704	0.000000	0.233252	0.000111	0.000238	0.000000	0.00000
_	2016-09-16	16:29:45.818510	CJSR	DFHSJITL		381	STC@CICS	DEVSYS01		10.018695	0.000000	10.018126	0.000064	0.000350	0.000000	0.00000
_	2016-09-16	16:29:45.676781	CWWU	DFHWBA	DFHWBBLI	380	EZRIEL	DEVSYS01		5.856291	0.000000	0.003808	0.000330	5.852482	0.000000	0.00000
_	2016-09-16	16:29:45.632369	CWWU	DFHWBA	DFHWBBLI	379	EZRIEL	DEVSYS01		0.015859	0.000000	0.002762	0.000043	0.013097	0.000000	0.00000
_	2016-09-16	16:29:43.341235	CWWU	DFHWBA	DFHWBBLI	378	EZRIEL	DEVSYS01		2.243498	0.000000	0.004658	0.000316	2.238840	0.000000	0.00000
_	2016-09-16	16:27:11.204158	CWWU	DFHWBA	DFHWBBLI	366	EZRIEL	DEVSYS01		0.007708	0.000000	0.002362	0.000041	0.005346	0.000000	0.00000
_	2016-09-16	16:27:11.194592	CWWU	DFHWBA	DFHWBBLI	365	EZRIEL	DEVSYS01		0.009100	0.000000	0.002832	0.000040	0.006268	0.000000	0.00000
_	2016-09-16	16:27:11.184599	CWWU	DFHWBA	DFHWBBLI	364	EZRIEL	DEVSYS01		0.007705	0.000000	0.002461	0.000041	0.005244	0.000000	0.00000
_	2016-09-16	16:27:11.173713	CWWU	DFHWBA	DFHWBBLI	363	EZRIEL	DEVSYS01		0.009156	0.000000	0.003642	0.000040	0.005514	0.000000	0.00000
_	2016-09-16	16:27:11.146767	CWWU	DFHWBA	DFHWBBLI	362	EZRIEL	DEVSYS01		0.009572	0.000000	0.002811	0.000061	0.006761	0.000000	0.00000
_	2016-09-16	16:25:28.386993	CJSA		DATABUS	361	CICSUSER	DEVSYS01		0.049273	0.000000	0.046848	0.000299	0.025698	0.000090	0.00000
_	2016-09-16	16:25:07.305304	CJSA		DATABUS	358	CICSUSER	DEVSYS01		0.933824	0.000000	0.919824	0.000286	0.140159	0.000095	0.00000
_	2016-09-16	16:25:05.793959	CJSU			357		DEVSYS01		0.273420	0.000000	0.272183	0.000123	0.001051	0.000000	0.00000
_	2016-09-16	16:25:05.793481	CJSU			356		DEVSYS01		0.161532	0.000000	0.158215	0.001560	0.003116	0.000000	0.00000
_	2016-09-16	16:24:55.472766	CJSR	DFHSJITL		354	STC@CICS	DEVSYS01		10.049841	0.000000	10.049413	0.000063	0.000291	0.000000	0.00000
_	2016-09-16	16:24:55.236390	CWWU	DFHWBA	DFHWBBLI	353	EZRIEL	DEVSYS01		6.094650	0.000000	0.003766	0.000371	6.090884	0.000000	0.00000
_	2016-09-16	16:24:55.203182	CWWU	DFHWBA	DFHWBBLI	352	EZRIEL	DEVSYS01		0.014675	0.000000	0.002757	0.000215	0.011917	0.000000	0.00000
_	2016-09-16	16:24:47.565550	CWWU	DFHWBA	DFHWBBLI	351	EZRIEL	DEVSYS01		7.586684	0.000000	0.004480	0.000329	7.582204	0.000000	0.00000
_	2016-09-16	16:24:09.320145	CWWU	DFHWBA	DFHWBBLI	350	EZRIEL	DEVSYS01		0.006873	0.000000	0.001947	0.000039	0.004925	0.000000	0.00000
_	2016-09-16	16:24:09.310729	CWWU	DFHWBA	DFHWBBLI	349	EZRIEL	DEVSYS01		0.006926	0.000000	0.001900	0.000036	0.005026	0.000000	0.00000
_	2016-09-16	16:24:09.301419	CWWU	DFHWBA	DFHWBBLI	348	EZRIEL	DEVSYS01		0.007031	0.000000	0.001987	0.000038	0.005043	0.000000	0.00000
_	2016-09-16	16:24:09.291097	CWWU	DFHWBA	DFHWBBLI	347	EZRIEL	DEVSYS01		0.006776	0.000000	0.002053	0.000042	0.004723	0.000000	0.00000
_	2016-09-16	16:24:09.276325	CWWU	DFHWBA	DFHWBBLI	346	EZRIEL	DEVSYS01		0.009826	0.000000	0.003158	0.000066	0.006667	0.000000	0.00000 ...

. . .

Profiler: Transaction List

- Displays only the transactions that meet your selection criteria
- Works well under pressure - supports viewing and scanning of very long transaction lists both efficiently and quickly.
- Commands:
  - **FORM**: Rearrange, omit, add display fields (columns)
  - **FILTER**: Change the selection criteria
  - **SORT**: Sort transactions by selected column (fast!)
  - **REFRESH**: Reapply selection criteria, check for new transactions

Tip: Use a wide screen to show many columns at once

C\PROF Transaction List Line 00000001
Command ===> Scroll ===> PAGE

/	Date	Time	Tran	Program	Second Program	Task Number	Userid	APPLID	ABEND Code	Response Time	First Dispatch Time	Application Processing Time	SYNCPOINT Time	EXEC CICS Time	VSAM File Time	Transien Dat Tim
_	2016-09-16	16:22:21.365961	CWWU	DFHWBA	DFHWBBLI	337	EZRIEL	DEVSYS01		5.832992	0.000000	0.003866	0.000388	5.829125	0.000000	0.00000
_	2016-09-16	16:22:21.332483	CWWU	DFHWBA	DFHWBBLI	336	EZRIEL	DEVSYS01		0.014784	0.000000	0.002628	0.000217	0.012155	0.000000	0.00000
_	2016-09-16	16:22:18.970160	CWWU	DFHWBA	DFHWBBLI	335	EZRIEL	DEVSYS01		2.300568	0.000000	0.005038	0.000284	2.295530	0.000000	0.00000
_	2016-09-16	16:08:57.104604	CJSA		DATABUS	292	CICSUSER	DEVSYS01		0.040158	0.000000	0.038035	0.000210	0.021617	0.000087	0.00000
_	2016-09-16	16:08:52.766728	CJSA		DATABUS	291	CICSUSER	DEVSYS01		0.054218	0.000000	0.051504	0.000208	0.027567	0.000098	0.00000
_	2016-09-16	16:08:48.074465	CJSA		DATABUS	290	CICSUSER	DEVSYS01		0.029168	0.000000	0.027031	0.000093	0.016647	0.000081	0.00000
_	2016-09-16	16:08:44.121332	CJSA		DATABUS	289	CICSUSER	DEVSYS01		0.868943	0.000000	0.866191	0.000169	0.037620	0.000147	0.00000

Profiler: Transaction List drilldown

Drill down into the details of a transaction using one of the following views:

- **Application Events (line action S)**
  - EXEC CICS, JCICS, DB2 SQL, IMS DLI, MQ, etc.
  - Response codes
  - Performance
  - Program links
  - Resource involvement
- **Application Commands (line action C)**
  - Significant events in the life of the transaction
- **Program Analysis (line action P)**
  - Graphical breakdown of transaction processing time by program and components
- **Transaction Overview (line action O)**
  - Transaction statistics summary
- **Trace Entries (line action T)**
  - Two sub-views: Short and Full

Profiler: Application Events (line action S)

```
C\PROF Application Events
Command ===> ____________________________________________________________

Tran: CWWU  Start: 2017-01-16 16:17:57.203173 Response: 0.019165   Task: 308

/    Relative    Elapsed Call                Command
__ __________ __________ __________________ ____________________________________________________
__   +0.000000            ATTACH TASK        ATTACH/OK TRANNUM(0000308C) CURRENT_ACTIVE(7) CUR
__   +0.001167            START PROGRAM      START_PROGRAM PROGRAM(DFHWBA) CEDF_STATUS(NOCEDF)
__   +0.001176   0.000013 HANDLE ABEND       HANDLE ABEND LABEL(X'28CFD54A') PLX STMT_#(000005
__   +0.001193   0.000016 LINK               LINK PROGRAM('DFHWBBLI') COMMAREA('.Q>DFHWBBLIPAR
__   +0.001227   0.000012 HANDLE ABEND       HANDLE ABEND LABEL(X'290DBB78') PLX STMT_#(000007
__   +0.001244   0.000012 LINK               LINK PROGRAM('DFHWUIPG') COMMAREA('GET /CICSSyste
__   +0.001262   0.000012 HANDLE ABEND       HANDLE ABEND LABEL(X'2A00343A') PLX STMT_#(000008
__   +0.001274   0.000008 ASSIGN             ASSIGN PLX STMT_#(00000922)
__   +0.001287   0.000017 LOAD               LOAD PROGRAM('DFHEITBS') SET(X'28D9C000') PLX STM
__   +0.001305   0.000012 LOAD               LOAD PROGRAM('DFHEITAB') SET(X'28DBC300') PLX STM
__   +0.001318   0.000012 LOAD               LOAD PROGRAM('DFHEITSZ') SET(X'28AFCB00') PLX STM
__   +0.001330   0.000029 WEB READ           WEB READ HTTPHEADER('User-Agent') NAMELENGTH(10)
__   +0.001360   0.000029 WEB EXTRACT        EXTRACT WEB HTTPMETHOD(X'C7C5E3') METHODLENGTH(3)
__   +0.001398   0.000008 ADDRESS            ADDRESS PLX STMT_#(00000560)
__   +0.001489   0.000007 ADDRESS            ADDRESS PLX STMT_#(00000298)
__   +0.001514   0.000013 HANDLE ABEND       HANDLE ABEND LABEL(X'292A8E94') ASM
__   +0.001531   0.000020 GETMAIN            GETMAIN SET(X'292AD448') FLENGTH(16380) INITIMG(X
```

Profiler: Application Events (line action S)

- Shows events in the life of the transaction
  - EXEC CICS, JCICS, DB2 SQL, IMS DLI, MQ, etc.
  - Response codes
  - Performance
  - Program links
  - Resource involvement
- To view associated trace entries, use line action S for a single entry, or use SS to mark out a block
- Choose from the following columns using `FORM` command:

Relative time	APPLID	Task	Program	Elapsed Time
Call	Resource	EIBRESP	EIBRESP2	Command

Profiler: Application Events (line action S)

Select as few or as many columns as you like:

```
C\PROF Application Events                                                    Row 51 of 891 More: >
Command ===> ____________________________________________________________  Scroll ===> PAGE

Tran: BPMT  Start: 2016-10-25 16:32:43.946398 Response: 0.096274   Task: 1106

/    Relative APPLID   Program     Elapsed Call          Resource  EIBRESP EIBRESP2 Command
__  +0.000338 FUWFWAR  MBKPCOM1  0.000016 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 UMBPTC BPMT') LENGTH(
__  +0.000367 FUWFWAR  MBKPCOM1  0.000007 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 UMBUSR PAYMENT') LENG
__  +0.000380 FUWFWAR  MBKPCOM1  0.000007 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 ++ Called KOCRMCLL ++
__  +0.000394 FUWFWAR  MBKPCOM1  0.000005 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 MBKPSEQ1 started') LE
__  +0.000404 FUWFWAR  MBKPCOM1  0.001984 LINK          MBKPSEQ1  OK      0        LINK PROGRAM('MBKPSEQ1') COMMAREA('0002000') LENGTH(7) COBOLII STMT_#(00117)
__  +0.002401 FUWFWAR  MBKPCOM1  0.000011 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 WS-SEQ 0002000') LENG
__  +0.002419 FUWFWAR  MBKPCOM1  0.000007 LINK          MBKPDEB1                   LINK PROGRAM('MBKPDEB1') COMMAREA(' 1234567890 0000000000000100{PAYMENT TEST
__  +0.002447 FUWFWAR  MBKPDEB1  0.000005 GETMAIN                  OK      0        GETMAIN SET(X'24C22408') FLENGTH(4304) SYSEIB ASM STMT_#(00000460)
__  +0.002464 FUWFWAR  MBKPDEB1  0.000512 READ          MBKACCT1  NOTFND  80       READ FILE('MBKACCT1') INTO('... ...') LENGTH(207) RIDFLD('1234567890') KEYLE
__  +0.002980 FUWFWAR  MBKPDEB1  0.000030 LINK          DFHDYP                     LINK PROGRAM('MBKPDLCE') COMMAREA('COBEDPA02106B1293P009 RESISTOR ...') LENG
__  +0.003019 FUWFWAR  DFHDYP              RETURN                *DHTJ*           RETURN ASM
__  +0.041909 FUWFWAR  DFHDYP   0.000016 GETMAIN                  OK      0        GETMAIN SET(X'24C234E8') FLENGTH(14904) SYSEIB ASM STMT_#(00000460)
__  +0.041930 FUWFWAR  DFHDYP   0.000004 GETMAIN                  OK      0        GETMAIN SET(X'24C26F38') FLENGTH(4080) SYSEIB ASM STMT_#(00000460)
__  +0.041942 FUWFWAR  DFHDYP   0.000003 GETMAIN                  OK      0        GETMAIN SET(X'24C27F38') FLENGTH(7232) SYSEIB ASM STMT_#(00000460)
__  +0.041958 FUWFWAR  DFHDYP   0.000003 GETMAIN                  OK      0        GETMAIN SET(X'24C29B88') FLENGTH(4080) SYSEIB ASM STMT_#(00000460)
__  +0.041967 FUWFWAR  DFHDYP   0.000003 ADDRESS                  OK      0        ADDRESS SYSEIB ASM STMT_#(00000229)
__  +0.041973 FUWFWAR  DFHDYP   0.000002 GETMAIN                  OK      0        GETMAIN SET(X'24C2AB88') FLENGTH(472) USERDATAKEY SYSEIB ASM STMT_#(00000279
__  +0.041979 FUWFWAR  DFHDYP   0.000003 GETMAIN                  OK      0        GETMAIN SET(X'24C2AD78') FLENGTH(4080) SYSEIB ASM STMT_#(00000460)
__  +0.041994 FUWFWAR  DFHDYP   0.000002 ADDRESS                  OK      0        ADDRESS SYSEIB ASM STMT_#(00000229)
__  +0.041998 FUWFWAR  DFHDYP   0.000003 GETMAIN                  OK      0        GETMAIN SET(X'24C2BD78') FLENGTH(648) USERDATAKEY SYSEIB ASM STMT_#(00000279
__  +0.042026 FUWFWAR  DFHDYP   0.000002 ADDRESS                  OK      0        ADDRESS SYSEIB ASM STMT_#(00000247)
__  +0.042031 FUWFWAR  DFHDYP   0.000015 LOAD          CEEMENU3  OK      0        LOAD PROGRAM('CEEMENU3') SET(X'247A2760') FLENGTH(54416) ENTRY(X'A47A2760')
__  +0.042061 FUWFWAR  DFHDYP   0.000085 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 CEE3250C The system o
__  +0.042163 FUWFWAR  DFHDYP   0.000007 WRITEQ TD     CESE      OK      0        WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 From compile unit MBK
```

Profiler: Application Events (line action S)

Reveal the underlying activity behind Java JCICS calls:

```
C\PROF Application Events                                                                    Row 1 of 21 More: >
Command ===> _____________________________________________________________________________ Scroll ===> PAGE

Tran: CJSA  Start: 2017-02-20 13:08:32.709045 Response: 0.016944   Task: 129

/    Relative Program     Elapsed Call          Resource EIBRESP Command
__ __________ _______ __________ ____________ ________ _______ ____________________________________________________________________________
__  +0.000000                    ATTACH TASK                    BUILD_TRANSACTION/OK TRANNUM(0000129C)
__  +0.000208           0.000355 JCICS                          DTCTask_getCommonData
__  +0.000624           0.000059 JCICS                          DTCTerminal_getUserId
__  +0.002796           0.000474 JCICS                          enterTrace
__  +0.007704           0.000310 JCICS                          enterTrace
__  +0.008053           0.000136 JCICS                          Task_GETCURRENTCHANNEL
__  +0.008395           0.000336 JCICS                          Container_PUT
__  +0.008763                    JCICS                          DTCProgram_LINK envp(000000004FC73200)
__  +0.008788           0.000072 LINK          DATABUS  OK      LINK
__  +0.009100 DATABUS    0.000021 ASSIGN                 OK      ASSIGN COBOLII STMT_#(00053)
__  +0.009122 DATABUS    0.000064 GET CONTAINER          OK      GET CONTAINER('request-cont') CHANNEL('payroll') INTO(AT X'2960D1A0') FLENGTH(10) COBOLII STMT
__  +0.009199 DATABUS    0.000149 READ          PAYROLL  OK      READ FILE('PAYROLL') INTO('100001CIRCLE SOFTWARE 35 WATERVIEW BLVD PARSIPPANY-TROY HILLS, NJ 0
__  +0.009348 DATABUS    0.000033 PUT CONTAINER          OK      PUT CONTAINER('request-cont') CHANNEL('payroll') FROM(AT X'2960D1A0') FLENGTH(576) COBOLII STM
__  +0.009382 DATABUS    0.007548 RETURN                         RETURN COBOLII STMT_#(00627)
__  +0.009562           0.000068 JCICS                          Container_GETLENGTH
__  +0.009647           0.000090 JCICS                          Container_GET
__  +0.009763           0.000102 JCICS                          enterTrace
__  +0.009882           0.000106 JCICS                          Channel_DELETE
__  +0.014299           0.000103 JCICS                          enterTrace
__  +0.016212           0.000325 JCICS                          DTC_Clean
__  +0.016944                    RELEASE TASK                   RELEASE_XM_CLIENT TERMINATION_TYPE(NORMAL)
```

Profiler: Application Commands (line action C)

- Review calls for selected transaction
- Formatted results
- Easier to read
- See the calls without the noise
- Scroll down to see more

```
C\PROF Application Commands
Command ===> ____________________________________________________________________

   Tran: BPMT Task: 1106   Start: 2016-10-25 16:32:43.946398 Response: 0.096274
Program: MBKPSTD1          Event: 2016-10-25 16.32.43.946398 Relative: 0.000000
 STMT_#:                EIBRESP:                               Elapsed:

Command
--------------------------------------------------------------------------------
ATTACH/OK TRANSACTION_TOKEN(2384F100 , 0001106C) TRANNUM(0001106C)

START_PROGRAM PROGRAM(MBKPSTD1) CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI) ENVIRONMENT_TYPE(EXEC) SYNCONRETURN(NO) LANGUAGE_BLOCK(
              00000000) LINK_LEVEL(1) SYSEIB_REQUEST(NO)

GETMAIN SET(X'24C0D828') FLENGTH(4256) SYSEIB ASM STMT_#(00000460)

LINK PROGRAM('MBKPCOM1') COMMAREA('BPMT 1234567890 2234567890 0000000000000100{0000000000000100{PA') LENGTH(282) COBOLII STMT_#(

GETMAIN SET(X'24C17498') FLENGTH(5080) SYSEIB ASM STMT_#(00000460)

GETMAIN SET(X'24C18888') FLENGTH(8) INITIMG(X'40') COBOLII STMT_#(00141)

MONITOR POINT(2) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00147)

MONITOR POINT(1) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00153)

FREEMAIN DATAPOINTER(X'24C18888') COBOLII STMT_#(00159)

GETMAIN SET(X'24C18888') FLENGTH(4080) SYSEIB ASM STMT_#(00000460)

WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 UMBPTC BPMT') LENGTH(36) SYSEIB ASM STMT_#(00000412)

WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 UMBUSR PAYMENT') LENGTH(40) SYSEIB ASM STMT_#(00000412)

WRITEQ TD QUEUE('CESE') FROM(' UW3BBPMT 20161025163243 ++ Called KOCRMCLL ++') LENGTH(46) SYSEIB ASM STMT_#(00000412)
```

Profiler: Application Commands (line action C)

- Shows events in the life of the transaction
- Easy to read format
- Simplified to minimize distraction
- Review just the application calls
- There is a limit of approximately 100 bytes of information per call, therefore COMMAREA and I/O areas may be truncated.
- You can scroll up and down to view all the events, or quickly switch back to “transaction detail” mode to see the call latencies.

Profiler: Program Analysis (line action P)

```
C\PROF Program Analysis
Command ===> ____________________________________________________________

Tran: BPMT  Start: 2016-11-24 08:23:37.920728 Response: 0.036154   Task: 1476

                          Percent of Processing Time
  Program      Elapsed        *----1----2----3----4----5----6----7----8----9----*
- MBKPSTD1    0.000211    0.58
  Process     0.000182    0.50
  PC          0.000012    0.04
  SM          0.000016    0.05
- MBKPCOM1    0.000783    2.17 ■
  Process     0.000073    0.20
  PC          0.000595    1.65 ■
  TD          0.000070    0.20
  SM          0.000029    0.08
  Other       0.000014    0.04
- MBKPDEB1    0.000192    0.53
  Process     0.000017    0.05
  PC          0.000033    0.09
  FC          0.000133    0.37
  SM          0.000008    0.02
- DFHDYP      0.000268    0.74
  Process     0.000248    0.69
  PC          0.000019    0.05
- MBKPDLCE    0.034694   95.97 ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
  Process     0.028585   79.07 ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
  PC          0.000058    0.16
  TD          0.005759   15.93 ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
  SM          0.000242    0.67
  ENQ/DEQ     0.000009    0.03
  IMS         0.000007    0.02
  Other       0.000030    0.09
```

- Breakdown of processing by program and component
- Programs listed in order of use, but if called multiple times the values are accumulated and presented once only.
- Expand/collapse a program to show/hide

Profiler: Program Analysis (line action P) - components

Component	Meaning
Process	Application code has control
BA	Business Activity
DB2	DB2 SQL
DH	Document Handler
ENQ	Enqueue/Dequeue
EP	Event Publishing
FC	File Control
IC	Interval Control
IMS	IMS-DBCTL DLI
JCICS	Java JCICS requests
LG	Logger Journalling
MQ	IBM MQ

Component	Meaning
NC	Named Counters
Other	All other EXEC CICS command types
PC	Program Load
SM	Storage Manager
Syncpoint	EXEC CICS SYNCPOINT and commit during return
TC	Terminal Control and BMS
TD	Transient Data
TS	Temporary Storage
WEB	WEB and services
XML	XML and JSON conversion
Unknown	Unaccountable time

Profiler: Performance summary (line action O)

```
C\PROF Transaction Overview
Command ===> _____________________________________

Date  . . . . . . . . . . . . :          2016-11-24
Time  . . . . . . . . . . . . :       08:23:37.920728
Tran  . . . . . . . . . . . . :                 BPMT
Program . . . . . . . . . . . :             MBKPSTD1
Second Program  . . . . . . . :             MBKPCOM1
Task Number . . . . . . . . . :                 1476
Userid  . . . . . . . . . . . :             CICSUSER
Group . . . . . . . . . . . . :                 JOHN
APPLID  . . . . . . . . . . . :             FUWFWAR
ABEND Code  . . . . . . . . . :                 DHTJ
Response Time . . . . . . . . :             0.036154
Application Processing Time . :             0.029106
EXEC CICS Time  . . . . . . . :             0.007035
VSAM File Time  . . . . . . . :             0.000133
Transient Data Time . . . . . :             0.005830
Program Call Time . . . . . . :             0.000719
Interval Control Time . . . . :             0.000297
ENQ/DEQ Time  . . . . . . . . :             0.000009
IMS Time  . . . . . . . . . . :             0.000007
Dispatch Count  . . . . . . . :                  426
EXEC CICS Count . . . . . . . :                  420
VSAM File Count . . . . . . . :                    1
Transient Data Count  . . . . :                  370
Program Call Count  . . . . . :                   10
Interval Control Count  . . . :                   28
ENQ/DEQ Count . . . . . . . . :                    1
IMS Count . . . . . . . . . . :                    1
Network Name  . . . . . . . . :          DEV1.VAPFUW3B
Network UOW . . . . . . . . . :       B114D06C03FE0001
Recovery UOW  . . . . . . . . :       D1B114D06C11D02A
Job Name  . . . . . . . . . . :               FUWFWAR
CICS VRM  . . . . . . . . . . :                  0700
```

Critical identification and performance information for a single transaction all in one place

```
C\PROF Transaction Overview
Command ===> _____________________________________

Date  . . . . . . . . . . . . :          2017-02-20
Time  . . . . . . . . . . . . :       13:08:32.709045
Tran  . . . . . . . . . . . . :                 CJSA
Second Program  . . . . . . . :              DATABUS
Task Number . . . . . . . . . :                  129
Userid  . . . . . . . . . . . :             CICSUSER
APPLID  . . . . . . . . . . . :              DEV1234
Response Time . . . . . . . . :             0.016944
Application Processing Time . :             0.014260
SYNCPOINT Time  . . . . . . . :             0.000141
EXEC CICS Time  . . . . . . . :             0.009898
VSAM File Time  . . . . . . . :             0.000149
Program Call Time . . . . . . :             0.007930
Business Activity Time  . . . :             0.000579
JCICS Time  . . . . . . . . . :             0.002470
Dispatch Count  . . . . . . . :                   33
EXEC CICS Count . . . . . . . :                   18
VSAM File Count . . . . . . . :                    1
Program Call Count  . . . . . :                    3
Business Activity Count . . . :                    6
JCICS Count . . . . . . . . . :                   13
Network Name  . . . . . . . . :
Network UOW . . . . . . . . . :  0000000000000000
Recovery UOW  . . . . . . . . :  0000000000000000
Job Name  . . . . . . . . . . :              DEV1234
CICS VRM  . . . . . . . . . . :                 0700
```

Profiler: Trace events (deep dive analysis - short)

```
C\PROF Trace Events
Command ===>
   16:32:43.946398 Level 3  Gap < 000000.008 >
/  Time (LOCAL)
__ 16:32:43.946398 XM 1102 XMAT  EXIT  ATTACH/OK TRANSACTION_TOKEN(2384F100 , 0001106C) TRANNUM(0001106C)
__ 16:32:43.946469 RM FA01 RMUC  ENTRY CREATE_UOW UOW_ID(2384F1A0 , 0000001B) HEURISM(NO) CHOICE(BACKWARD) INDOUBT_TIMEOUT_IN
__ 16:32:43.946490 XS 070A XSRC  EVENT CHECK-COMPLETE BPMT CICSUSER FUNCTION(CHECK_RESOURCE_ACCESS) RESPONSE(OK) SAF_RESPONSE
                                       ESM_REASON(0)
__ 16:32:43.946513 AP 2520 ERM   ENTRY CALL-TRUES-FOR-TASK-START
__ 16:32:43.946527 PG 0901 PGPG  ENTRY INITIAL_LINK PROGRAM_NAME(MBKPSTD1)
__ 16:32:43.946533 LD 0002 LDLD  EXIT  ACQUIRE_PROGRAM/OK ENTRY_POINT(A3EF0EA0) LOAD_POINT(23EF0000) PROGRAM_LENGTH(86E8) PRO
                                       LOCATION(ERDSA) COPY_STATUS(OLD_COPY) FETCH_TIME() LIBRARY(DFHRPL) BUNDLE_INSTALLED_LI
__ 16:32:43.946537 AP 1940 APLI  ENTRY START_PROGRAM PROGRAM(MBKPSTD1) CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI) ENVIRONMENT_T
                                       LANGUAGE_BLOCK(23A926B8) COMMAREA(00000000 , 00000000) LINK_LEVEL(1) SYSEIB_REQUEST(NO
__ 16:32:43.946605 AP E160 EXEC  ENTRY GETMAIN SET(AT X'24C00110') FLENGTH(4256) SYSEIB ASM STMT_#(00000460)
__ 16:32:43.946621 AP E161 EXEC  EXIT  GETMAIN SET(X'24C0D828') FLENGTH(4256) SYSEIB ASM STMT_#(00000460)
__ 16:32:43.946634 AP E160 EXEC  ENTRY LINK PROGRAM('MBKPCOM1') COMMAREA('BPMT 1234567890 2234567890 0000000000000100{0000000
                                       STMT_#(00065)
__ 16:32:43.946638 PG 1101 PGLE  ENTRY LINK_EXEC PROGRAM_NAME(MBKPCOM1) COMMAREA(24C0D8F0 , 0000011A) SYSEIB_REQUEST(NO) FORC
__ 16:32:43.946649 AP 1940 APLI  ENTRY START_PROGRAM PROGRAM(MBKPCOM1) CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI) ENVIRONMENT_T
                                       LANGUAGE_BLOCK(23A92704) COMMAREA(24C0D8F0 , 0000011A) LINK_LEVEL(2) SYSEIB_REQUEST(NO
__ 16:32:43.946666 AP E160 EXEC  ENTRY GETMAIN SET(AT X'24C00110') FLENGTH(5080) SYSEIB ASM STMT_#(00000460)
__ 16:32:43.946670 AP E161 EXEC  EXIT  GETMAIN SET(X'24C17498') FLENGTH(5080) SYSEIB ASM STMT_#(00000460)
__ 16:32:43.946678 AP E160 EXEC  ENTRY GETMAIN SET(AT X'24C175E8') FLENGTH(8) INITIMG(X'40') COBOLII STMT_#(00141)
__ 16:32:43.946682 AP E161 EXEC  EXIT  GETMAIN SET(X'24C18888') FLENGTH(8) INITIMG(X'40') COBOLII STMT_#(00141)
__ 16:32:43.946685 AP E160 EXEC  ENTRY MONITOR POINT(2) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00147)
__ 16:32:43.946693 AP E161 EXEC  EXIT  MONITOR POINT(2) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00147)
__ 16:32:43.946695 AP E160 EXEC  ENTRY MONITOR POINT(1) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00153)
__ 16:32:43.946698 AP E161 EXEC  EXIT  MONITOR POINT(1) DATA1('.Ahh') DATA2('...') ENTRYNAME('DFHAPPL') COBOLII STMT_#(00153)
__ 16:32:43.946701 AP E160 EXEC  ENTRY FREEMAIN DATAPOINTER(X'24C18888') COBOLII STMT_#(00159)
__ 16:32:43.946705 AP E161 EXEC  EXIT  FREEMAIN DATAPOINTER(X'24C18888') COBOLII STMT_#(00159)
```

```
C\PROF Trace Events                                    Record 00000001 More: < >
Command ===> ____________________________________________ Scroll ===> CSR
   13:07:50.575308 Level 0  Gap < 000000.008 >           Format ===> WRAP
/  Time (LOCAL)    ------------------------------------------------------------
__ 13:07:50.575308 AP 21E0 JCICS ENTRY DTCProgram_LINK envp(000000004FD83300)
__ 13:07:50.575330 AP 00E1 EIP   ENTRY LINK REQ(0004)
__ 13:07:50.575347 PG 1101 PGLE  ENTRY LINK_EXEC PROGRAM_NAME(DATABUS)
                   SYSEIB_REQUEST(NO) FORCE_LOCAL(NO) CHANNEL(payroll)
__ 13:07:50.575386 LD 0002 LDLD  EXIT  ACQUIRE_PROGRAM/OK ENTRY_POINT(A9413028)
                   LOAD_POINT(29413000) PROGRAM_LENGTH(35C8)
                   PROGRAM_ATTRIBUTE(REUSABLE) LOCATION(ESDSA)
                   COPY_STATUS(OLD_COPY) FETCH_TIME() LIBRARY(CICSWS)
                   BUNDLE_INSTALLED_LIB(NO) PRIVATE_LIBRARY(NO)
__ 13:07:50.575407 AP 1940 APLI  ENTRY START_PROGRAM PROGRAM(DATABUS)
                   CEDF_STATUS(CEDF) EXECUTION_SET(FULLAPI)
                   ENVIRONMENT_TYPE(EXEC) SYNCONRETURN(NO)
                   LANGUAGE_BLOCK(29106328) COMMAREA(00000000 , 00000000)
                   LINK_LEVEL(2) SYSEIB_REQUEST(NO)
__ 13:07:50.575604 AP 00E1 EIP   ENTRY ASSIGN REQ(0004)
__ 13:07:50.575616 AP E160 EXEC  ENTRY ASSIGN COBOLII STMT_#(00053)
__ 13:07:50.575625 AP E161 EXEC  EXIT  ASSIGN COBOLII STMT_#(00053)
__ 13:07:50.575625 AP 00E1 EIP   EXIT  ASSIGN OK REQ(00F4)
__ 13:07:50.575629 AP 00E1 EIP   ENTRY GET-CONTAINER REQ(0004)
__ 13:07:50.575638 AP E160 EXEC  ENTRY GET CONTAINER('request-cont')
                   CHANNEL('payroll') INTO(AT X'2960D1A0') FLENGTH(576) COBOLII
                   STMT_#(00056)
__ 13:07:50.575693 AP E161 EXEC  EXIT  GET CONTAINER('request-cont')
                   CHANNEL('payroll') INTO(AT X'2960D1A0') FLENGTH(10) COBOLII
                   STMT_#(00056)
__ 13:07:50.575694 AP 00E1 EIP   EXIT  GET-CONTAINER OK REQ(00F4)
__ 13:07:50.575706 AP 00E1 EIP   ENTRY READ REQ(0004)
__ 13:07:50.575715 AP E160 EXEC  ENTRY READ FILE('PAYROLL') INTO(AT
                   X'2960D458') LENGTH(150) RIDFLD(AT X'A960D1A4') EQUAL
                   COBOLII STMT_#(00097)
__ 13:07:50.575856 AP 04F1 EIFC  EXIT  READ_FILE RESP=0 RESP2=0
__ 13:07:50.575865 AP E161 EXEC  EXIT  READ FILE('PAYROLL') INTO('100001CIRCLE
                   SOFTWARE 35 WATERVIEW BLVD PARSIPPANY-TROY HILLS, NJ 07054
```


Profiler:

Trace events (deep dive analysis - short)

Anatomy of a JCICS call

Profiler: Trace events (deep dive analysis)

- **Powerful trace viewer** used to display all captured events associated with…
  - a complete transaction
  - an individual application call within a transaction
  - an auxiliary trace data set (in full)
- Like the IBM Trace utility print program (DFHTU700) only **INTERACTIVE!**
- If using the **multiregion operation (MRO) option**, the display will contain trace entries from all participating CICS regions
- Fields to control and format the display:
  - **Time**: jump to particular time in trace
  - **Gap**: Highlight time gaps in the trace
  - **Level**: control point IDs included in display
  - **Format**: Display entries on a single line or use line wrapping

- **Multiple line actions and commands** to remove individual trace entries (`X`) and whole trace domains (`XD`), add filters (`FILTER`), find strings (`FIND`), change the time format (`TIME`), and track an individual transaction (show entries with same task number - `TX`)
- To view the full trace event and see its parameters, ***use line action s…***

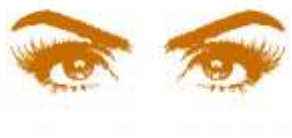
Profiler: Trace events (deep dive analysis - full)

```
BROWSE     SYS17051.T161229.RA000.USR.R0127286   Record 00000003 Line 00000000
Command ===> _____________________________________________ Scroll ===> PAGE
                                                          Format ===> STD
******************************** Top of data ********************************
+0020  Code... XS    070A
+003C  STCK... D18BCA2F251FAE1C      LSN.... 0000000000000003
       Date... 2016-10-25 Tuesday    Time... 16:32:43.946490.881

+0014  JOBN..... 'FUWFWAR '

+001C  TREN..... Trace entry
+0020  CALLER..... +23         POINTID.... 070A        TYPE..... E0
+0025  TASK..... +1106        KE_NUM..... 004C        OWNER..... 98D0
+002E  TCB..... 'QR   '      TCBADDR.... 008C7E88    RETADDR.... A225A78A

+0044  PARMs..... Parameters
+0046  PARM..... 1
       +0000  00600000 000000E8 00000000 00000000   *.-.....Y........*
       +0010  BFF40000 00000000 01000100 00000000   *.4..............*
       +0020  00000000 00000000 00000000 42601530   *.............-..*
       +0030  00000001 E3C3C9C3 E2E3D9D5 01010000   *....TCICSTRN....*
       +0040  23764F54 00000004 00000000 00000000   *..|.............*
       +0050  00000000 00000000 00000000 00000000   *................*
+00A8  PARM..... 2
       +0000  C2D7D4E3                             *BPMT            *
+00AE  PARM..... 3
       +0000  C3C9C3E2 E4E2C5D9                    *CICSUSER        *
******************************** End of data ********************************
```



Tip: Scrolling right or left will take you to the next or previous trace entry…

Primary Menu Option 3: Snap

Snap

```
                                    Snap
Command ===> ____________________________________________________________

1  Dynamic  New auxiliary data set is allocated on demand
2  Static   Use the auxiliary trace data set specified below

CICS region:
  Job name . . . . . . . MYCICSRG
  LPAR . . . . . . . . . PROD1

Auxiliary trace:
  Data set name  . . . . 'USR.SNAP.TXCAUXT'
  Space units  . . . . . MB        CYLS or MB
  Primary quantity . . . 32        In above units
  Secondary quantity . . 32        In above units
```

Capture the ***current*** contents of the CICS internal trace

- A “point-in-time” snapshot of what recently occurred in CICS
- Writes to an auxiliary trace data set

Snap

- A “point-in-time” snapshot of what recently occurred in CICS
  - The effect is that it can capture unexpected problems *when they occur* – *you will not need to reproduce the problem like you do with the CICS auxiliary trace.*
  - The bigger the internal trace table, the further you can “look back”…
  - *Very* lightweight, almost no CPU, completes quickly
- Snap writes to a standard **auxiliary trace data set**
  - Same format as IBM Trace utility print program (DFHTU700)
  - View with C\Prof *or* with DFHTU700 – same format!
  - A new capability that works with your existing process
- Capture **multiple regions at once** – just add the `CICS=` statement to the generated JCL

Primary Menu Option 4: Record

Record

```
                                    Record                          Row 1 to 1 of 1
Command ===> _____________________________________________________ Scroll ===> PAGE

1  Dynamic  New auxiliary data sets are allocated on demand
2  Static   Use the auxiliary trace data sets specified below

CICS region
  Job name . . . . . . . MYCICSRG
  LPAR . . . . . . . . . PROD1

New data set allocation attributes
  Space units  . . . . . CYLS      CYLS or MB
  Primary quantity . . . 30        In above units
  Secondary quantity . . 10        In above units

Static options
  Autoswitch (AUXTRSW) . ALL       (NO, NEXT or ALL)

/     Auxiliary trace data sets (Static): maximum of 32.
___   ____________________________________________________
```

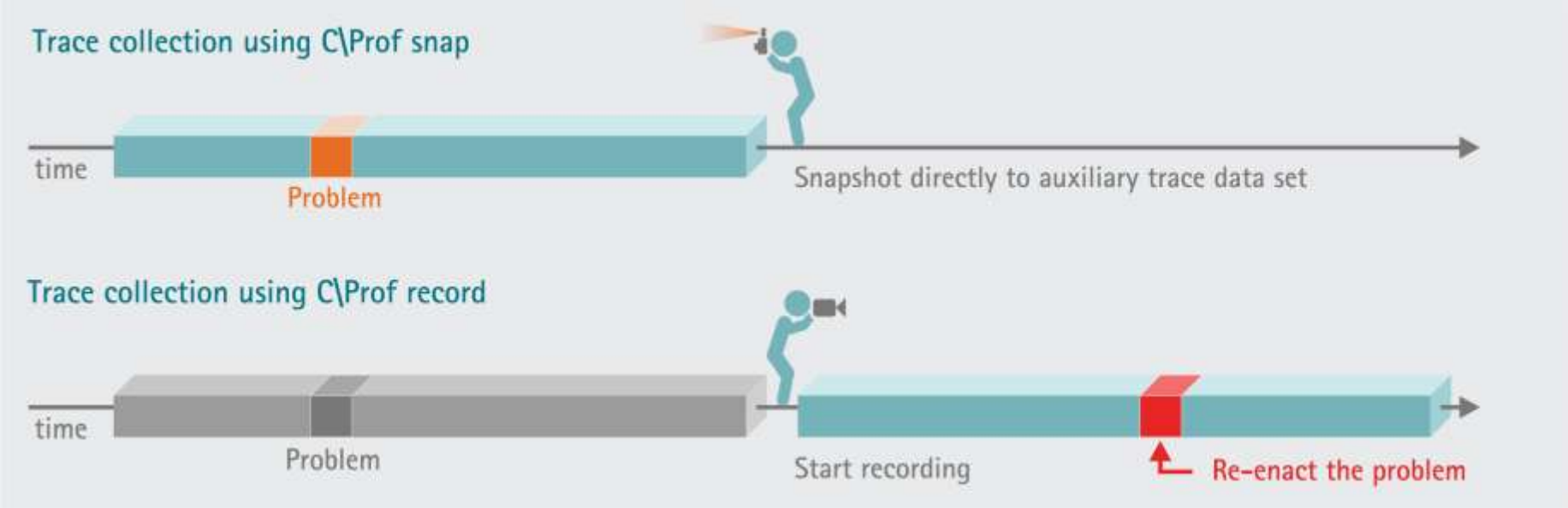
Capture the ***future*** contents of the CICS internal trace

- Similar to the CICS auxiliary trace
- Writes to auxiliary trace data sets

Record

- Record the future contents of the CICS internal trace to an auxiliary trace data set
- Like the CICS auxiliary trace, but:
  - Doesn't involve CICS (low overhead)
  - Has additional features to control data set creation and management
- Record from multiple regions in a single job by adding `CICS=` control statements

Snap vs Record: understanding the difference

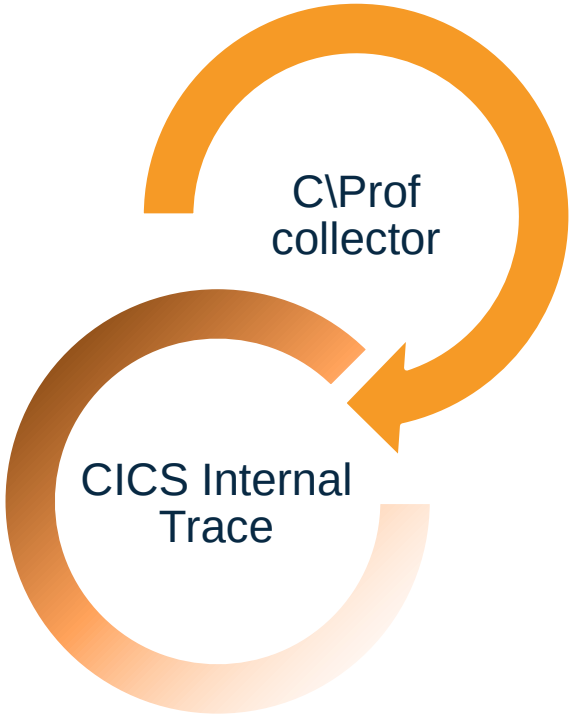


Load

- Load any auxiliary trace data set into the C\Prof profiler
- Pulls auxiliary trace data set event data into an archive trace data set
- View the application perspective (instead of just the trace events)
- Snap or Record, then Load
- Use the profiler on that old, unresolved problem you just can’t figure out!

Getting more (or less) from the trace

CICS internal trace: a rich source of information



- A “circular buffer” accessed by C\Prof from a separate address space
- CICS is completely unaware – no C\Prof code runs inside CICS!
- C\Prof must “keep up” with the trace and read it quickly before old trace entries are overwritten by new trace entries (wrapping)
  - C\Prof is in a race with CICS and must not get lapped!

CICS internal trace: trace point levels

- By default, C\Prof builds a **complete profile** of a transaction's activity using the trace point levels described in the table to the right.
- Trace points may be adjusted to provide **more or less** detail as required.
- If less detail is requested, C\Prof may produce a **partial profile**.
- Each additional trace level incurs additional overhead in CICS.
  - For example, trace level **EI 2** provides rich application call detail, but is more expensive than **EI 1** to collect.
  - The total additional overhead at higher levels may also depend on the *type of workload*.

Category	Level	Usage
AP	1	End of a transaction indicator JCICS, ABEND and other miscellaneous conditions
EI	2	EXEC CICS commands
LD and PG	1	Program related events including load
RA	1	IMS DBCTL
RI	1	DB2 and MQ
RM	1	Syncpoint commit and back-out Accounting network unit-of-work ID for MRO cross-region task correlation
XM	1	Start of a transaction indicator
XS	1	Security user ID

CICS internal trace: size, trace levels, and wrap speed

- The current **size** and **wrapping speed** of the CICS internal trace is reported during the discovery process:

```
TXC0382I CICS region is discovered: CICS TS V5.3 (700) Size=32768K Wrap=7.46 seconds
```

- To ensure that C\Prof can read the internal trace quickly enough:
  1. C\Prof requires the **same WLM service class as CICS**
  2. Set the trace table size to an **initial value of 32 MB** and increase it if C\Prof can't keep up.
     - Aim for a wrapping speed of **> 5 seconds**.
  3. Set your trace point levels accordingly:
     - For example, **reduce** the EXEC interface (EI) domain to 1 to sacrifice application call detail in favor of performance:

```
EI 1: WRITEQ-TD OK
EI 2: WRITEQ TD QUEUE('CESE') FROM('HELLO WORLD') LENGTH(11)
```

Configuration file control statements

Mode… …Keyword	COLLECT	RECORD (dynamic)	RECORD (static)	SNAP	LOAD	Description
CHECKPT	*Yes*	*Opt*	*Opt*	*No*	*Yes*	Checkpoint data set for archive registration
ARCHIVE	*Yes*	*Opt*	*Opt*	*No*	*Yes*	Archive data sets
SUMMARY	*Yes*	*No*	*No*	*No*	*Yes*	Summary archive data sets
DETAIL	*Yes*	*No*	*No*	*No*	*Yes*	Detail archive data sets
AUXILIARY	*No*	*Yes*	*No*	*No*	*No*	Auxiliary trace (archive) data sets
AUTOSTART	*Yes*	*Yes*	*Yes*	*No*	*No*	Automatically start collection
ACTIVATETRACE	*Yes*	*Yes*	*Yes*	*No*	*No*	Optimize CICS trace category levels
RESETTRACE	*Yes*	*Yes*	*Yes*	*No*	*No*	Reset CICS trace category levels
LEVEL	*Yes*	*No*	*No*	*No*	*Yes*	Level of trace detail to collect
STNTR*xx*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	Set domain trace point level
LIMIT	*Opt*	*Opt*	*Opt*	*No*	*No*	Time limit for collecting trace data (short burst)
GROUP	*Opt*	*Opt*	*Opt*	*Opt*	*Opt*	CICS region grouping for MRO and operations
CICS	*Yes*	*Yes*	*Yes*	*Yes*	*No*	CICS regions to monitor
MRO	*Opt*	*No*	*No*	*No*	*Opt*	MRO transaction task correlation
OUTPUT	*No*	*No*	*Yes*	*Yes*	*No*	Output to pre-defined auxiliary trace data sets
AUXTRSW	*No*	*No*	*Yes*	*No*	*No*	Autoswitch when auxiliary trace is full
INPUT	*No*	*No*	*No*	*No*	*Yes*	Input from auxiliary trace data sets

Configuration file example

```
COLLECT
LIMIT=1 HOUR

CHECKPT=CPROF.CHECKPT

ARCHDSN=CPROF.ARCH.+GROUP.+DATE.+TIME.+GEN
SUMMARY SPACE=(CYL,20)  UNIT=3390 RETPD=10 DAYS
DETAIL  SPACE=(CYL,100) UNIT=3390 RETPD=10 DAYS

RESPONSE=0.25
EXCLUDETRAN=(CICS)

AUTOSTART=1
ACTIVATETRACE=1
RESETTRACE=1
LEVEL=3

GROUP=CICSPROD MRO=YES
  CICS=CICSTOR
  CICS=CICSAOR
  CICS=CICSDOR
GROUPEND
```

Collect trace data for the purpose of profiling – shutdown after 1 hour

Checkpoint keeps a register of archive data sets created during collection

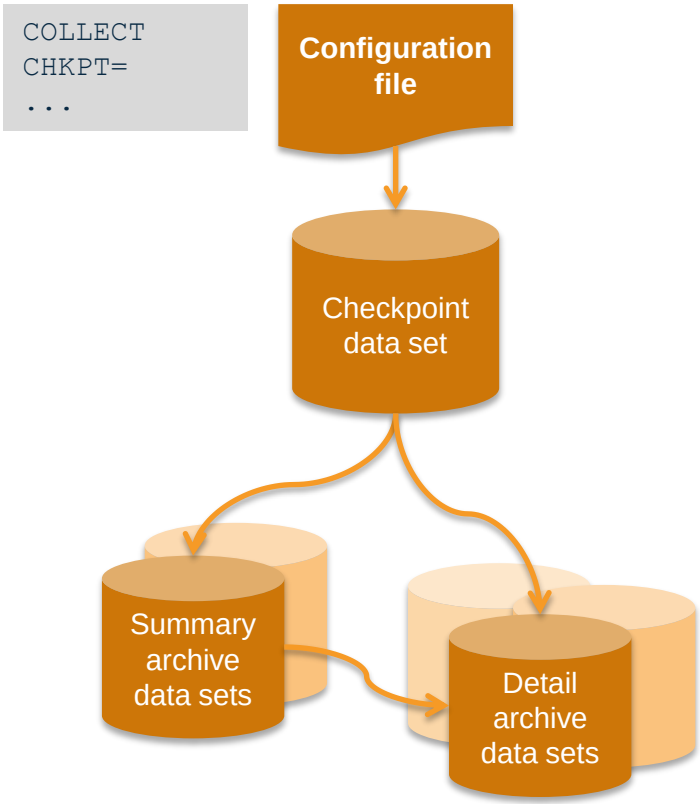
Archive data sets are allocated on demand

Filter options based on transaction code, program name, response time range, ABEND etc.

Control the monitoring of CICS by optionally setting of CICS trace category levels for optimal profiling

The CICS regions to monitor; optionally group them for MRO and operational management

Managing C\Prof archive data sets



- Used by the C\Prof collector to store data captured from the CICS internal trace
- Configuration file specifies the checkpoint data set to use (use `CHKPT=` to configure)
- Checkpoint data set (VSAM KSDS) acts as an index into the archive data sets:
  - Used by ISPF dialog to find data
  - Used by housekeeping task to cleanup expired data sets
- Two types:
  - **Summary**: High-level information
  - **Detail**: Call/trace events (deep dive)

Archive data set naming and storage options

- Two methods:
  - **GDG**: Uses generation data group
    - Automatic data housekeeping
    - GDG affords more control of the amount of data retained at any one time - you need to predefine the GDG base
  - **Dynamic**: Allocate on demand
    - Unlimited data sets
    - Ensure that you have sufficient DASD capacity available; quarantined if necessary with SMS rules
- Use configuration file control statement `ARCHDSN=` to configure

Statement	Method	Example
`ARCHDSN=CPROF.+CICS(+1)`	GDG	`CPROF.CICSAOR(+1)`
`ARCHDSN=CPROF.+CICS.+DATE+TIME.+GEN`	Dynamic	`CPROF.CICS1.D160624.T072347.S001`

Operational considerations and other features

- **Short burst collection**
  - Run in production for short periods
  - Specify a collection time limit in configuration file (`LIMIT=5 Minutes`)
  - Collector shuts down after limit reached

- **Collector control via operator commands**
  - Start and stop collection of individual regions or groups using MVS system commands

- **Multiregion operation (MRO)**
  - Instruct C\Prof to interpret a transaction distributed across multiple CICS regions (using MRO) as a single transaction

Creating and viewing your own trace entries

- Applications can write their own **custom trace entries** which can be viewed in C\Prof
- Use to add additional tracing, or to view tracing already present in your application program
- Configuration file control statements required:
  - `LEVEL=2`
  - `ACTIVATETRACE=1`, or use `STNTREI=1` and `STNTRAP=1` (or higher)
- Inserting a trace entry:
  - COBOL:

    ```
    EXEC CICS ENTER TRACENUM
    ```

  - Java:

    ```
    com.ibm.cics.server.EnterRequest
    ```

```
import com.ibm.cics.server.EnterRequest;
. . .
// Get the container data
Container contOut = chan.getContainer(containerNameOut);
byte[] response = contOut.get();

// Write a trace entry (example only)
EnterRequest er = new EnterRequest();
er.setException(true); // Application program exception
er.setResource("ABCDEF"); // 8 byte resource field
er.setTraceIdentifier((short)0102); // ID number
er.enterTrace(response); // Byte array
```

User trace entries displayed in the profiler...

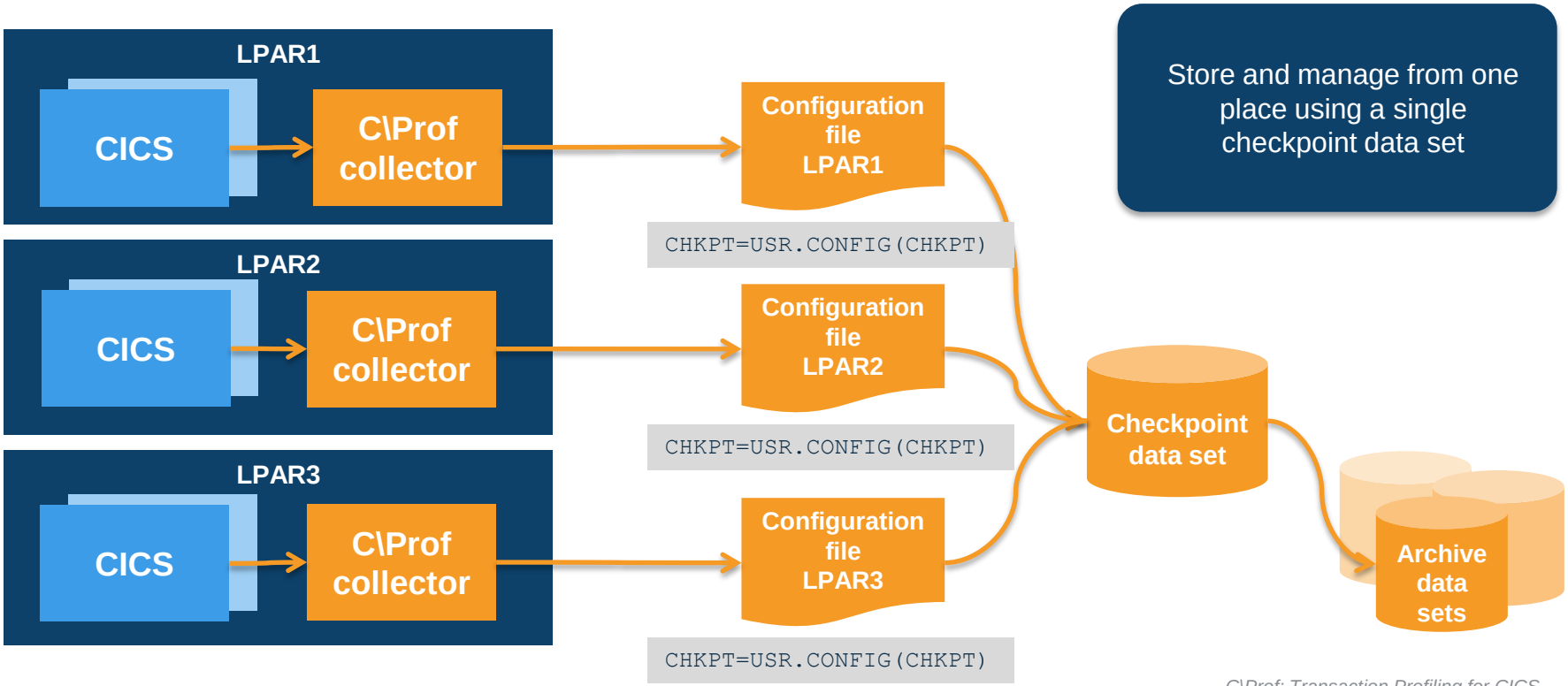
```
C\PROF Application Events                                    Row 1 of 21 More: >
Command ===> _____________________________________________ Scroll ===> PAGE

Tran: CJSA  Start: 2017-02-20 13:08:32.709045 Response: 0.016944   Task: 129

/    Relative Call          Resource Command
__ __________ ____________ ________ ____________________________________________
__  +0.000624 JCICS                 DTCTerminal_getUserId
__  +0.002796 JCICS                 enterTrace
__  +0.007704 JCICS                 enterTrace
__  +0.008053 JCICS                 Task_GETCURRENTCHANNEL
__  +0.008395 JCICS                 Container_PUT
__  +0.008763 JCICS                 DTCProgram_LINK envp(000000004FC73200)
__  +0.008788 LINK           DATABUS  LINK
__  +0.009100 ASSIGN                 ASSIGN COBOLII STMT_#(00053)
__  +0.009122 GET CONTAINER          GET CONTAINER('request-cont') CHANNEL('pay
__  +0.009199 READ           PAYROLL  READ FILE('PAYROLL') INTO('100001CIRCLE SO
__  +0.009348 PUT CONTAINER          PUT CONTAINER('request-cont') CHANNEL('pay
__  +0.009382 RETURN                 RETURN COBOLII STMT_#(00627)
__  +0.009562 JCICS                  Container_GETLENGTH
__  +0.009647 JCICS                  Container_GET
__  +0.009763 JCICS                  enterTrace
__  +0.009882 JCICS                  Channel_DELETE
__  +0.014299 JCICS                  enterTrace
__
```

- A Java program writing to the trace is shown as a JCICS call (command `enterTrace`)
- Drill down to the trace entries to see more

Collecting from multiple LPARs



C\Prof case study:

Profiling transactions where Java calls legacy business Cobol applications

The Transaction List shows the Java transactions and includes the program and EXEC CICS and SQL overhead of the AOR

```
C\PROF Transaction List
Command ===> _____
```

/	Time	Tran	Program	Task Number	Userid	Response Time	Application Processing Time	Program Call Count	Program Call Time	
_	14:58:45.579007	MOBJ	DFHSJTHP	48775	JOHN	0.473822	0.330601	27	0.101518	
_	14:58:42.890579	MOBJ	DFHSJTHP	48772	JOHN	0.318868	0.178353	27	0.066741	
_	14:58:41.153057	MOBJ	DFHSJTHP	48770	JOHN	0.319833	0.218369	27	0.075459	
_	14:58:38.056913	MOBJ	DFHSJTHP	48766	LYNDA	2.559088	0.256212	34	0.098705	. . .
_	14:58:37.352785	MOBJ	DFHSJTHP	48764	JOHN	1.519946	0.623545	27	0.074959	
_	14:58:37.193604	MOBJ	DFHSJTHP	48763	LYNDA	0.872566	0.493528	20	0.352856	
_	14:58:32.116551	MOBJ	DFHSJTHP	48758	LYNDA	4.674909	1.833652	144	0.406167	
_	14:58:26.932500	MOBJ	DFHSJTHP	48752	LYNDA	4.648170	1.698351	161	0.238702	

	DB2 Time	JCICS Time	EXEC CICS Count	EXEC CICS Time	Transient Data Time	Virtual Storage Time
	0.045989	0.027817	123	0.142204	0.001463	0.007023
	0.026648	0.006686	123	0.154562	0.002128	0.031446
	0.028726	0.014661	123	0.127525	0.001476	0.017761
. . .	2.093305	0.029021	167	0.260299	0.000007	0.101327
	0.693809	0.045806	123	0.255378	0.002005	0.019175
	0.187321	0.018009	93	0.492510	0.000000	0.051800
	2.121966	0.005785	582	0.843764	0.008619	0.303684
	2.049272	0.037753	657	0.938605	0.032507	0.371902

The Java owning region (JOR) runs applications that use JCICS and LINK to a COBOL program in the AOR to perform the business logic

How the transaction appears to the Java developer…

```
C\PROF Application Events                                                                    Row 1 of 29 More: >
Command ===>                                                                                 Scroll ===> CSR

Tran: PRMJ  Start: 2017-01-20 14:58:38.056913 Response: 2.559088   Task: 48766

/    Relative APPLID   Program     Elapsed Call          Resource  EIBRESP EIBRESP2 Command
__   +0.000000 CICSJOR  DFHSJTHP           ATTACH TASK                              ATTACH/OK TRANNUM(0048766C)
__   +0.024971 CICSJOR  DFHSJTHP  0.000119 JCICS                                    DTCTask_getCommonData
__   +0.025098 CICSJOR  DFHSJTHP  0.000010 JCICS                                    DTCTask_GETPROGRAM
__   +0.025330 CICSJOR  DFHSJTHP  0.000012 JCICS                                    Wrapper_GetCommArea envp(000000004C9B2100)
__   +0.034056 CICSJOR  DFHSJTHP  0.000012 JCICS                                    Task_GETCURRENTCHANNEL
__   +0.034087 CICSJOR  DFHSJTHP  0.023157 JCICS                                    Container_PUT
__   +0.057284 CICSJOR  DFHSJTHP  0.001345 JCICS                                    Container_PUT
__   +0.058655 CICSJOR  DFHSJTHP  0.000027 JCICS                                    Container_PUT
__   +0.065610 CICSJOR  DFHSJTHP           JCICS                                    DTCProgram_LINK envp(000000004C9B2100)
__   +0.065638 CICSJOR  DFHSJTHP  0.001540 LINK          EYU9XLOP  OK      0        LINK
__   +0.067205 CICSJOR  EYU9XLOP  0.001646 ASSIGN                    OK      0        ASSIGN OK
__   +0.068853 CICSJOR  EYU9XLOP  0.000010 HANDLE ABEND              OK      0        HANDLE-ABEND OK
__   +0.068948 CICSJOR  EYU9XLOP  0.010467 RETURN                                    SET_UOW/OK
__   +2.442180 CICSJOR  EYU9XLOP           START PROGRAM EYU9XLOP                   START_PROGRAM PROGRAM(EYU9XLOP) CEDF_STATUS(NOCEDF) EXEC_SET(FULLAPI)
__   +2.442237 CICSJOR  EYU9XLOP  0.000020 ASSIGN                    OK      0        ASSIGN OK
__   +2.445009 CICSJOR  EYU9XLOP  0.000011 HANDLE ABEND              OK      0        HANDLE-ABEND OK
__   +2.445362 CICSJOR  EYU9XLOP           RETURN                                   LINK OK
__   +2.453362 CICSJOR  DFHSJTHP  0.000024 JCICS                                    Container_GETLENGTH
__   +2.453395 CICSJOR  DFHSJTHP  0.000267 JCICS                                    Container_GET
__   +2.453704 CICSJOR  DFHSJTHP  0.000034 JCICS                                    Container_GETLENGTH
__   +2.453752 CICSJOR  DFHSJTHP  0.001958 JCICS                                    Container_GET
__   +2.455731 CICSJOR  DFHSJTHP  0.000019 JCICS                                    Container_GETLENGTH
__   +2.455762 CICSJOR  DFHSJTHP  0.002014 JCICS                                    Container_GET
__   +2.548949 CICSJOR  DFHSJTHP  0.000017 JCICS                                    DTC_Clean
__   +2.554923 CICSJOR  DFHSJTHP           START PROGRAM EYU9XLOP                   START_PROGRAM PROGRAM(EYU9XLOP) CEDF_STATUS(NOCEDF) EXEC_SET(FULLAPI)
__   +2.554941 CICSJOR  DFHSJTHP  0.000008 ASSIGN                    OK      0        ASSIGN OK
__   +2.556630 CICSJOR  DFHSJTHP  0.000010 HANDLE ABEND              OK      0        HANDLE-ABEND OK
__   +2.556749 CICSJOR  DFHSJTHP  0.002331 RETURN                                    COMMIT_UOW/OK FAILED_LINK(00000000)
__   +2.559088 CICSJOR  DFHSJTHP           RELEASE TASK                             RELEASE_XM_CLIENT TERMINATION_TYPE(NORMAL)
```

JCICS calls

JCICS calls

An unexplained delay of 2.5 seconds, waiting for the LINK to the AOR to respond

JCICS call: what happens under the covers?

```
C\PROF Trace Events                                                                    Record 00000001 More: < >
Command ===>                                                                            Scroll ===> CSR
   14:58:38.091000 Level 0  Gap < 4          >                                          Format ===> SNGL
/  Time (Relative) ------------------------------------------------------------------------------------------
__ 14:58:38.091000 AP 21E0 JCICS ENTRY Container_PUT
__       +0.000009 AP 00E1 EIP   ENTRY PUT-CONTAINER REQ(0004)
__       +0.000009 AP D500 UEH   EVENT LINK-TO-USER-EXIT-PROGRAM CMRXEIO3 AT EXIT POINT XEIIN
__       +0.000009 AP D501 UEH   EVENT RETURN-FROM-USER-EXIT-PROGRAM CMRXEIO3 WITH RETURN CODE 0
__       +0.000011 AP F801 EIBAM ENTRY PUT_CONTAINER
__       +0.000012 PG 1700 PGCH  ENTRY INQUIRE_CHANNEL CHANNEL_NAME(PRIME2)
__       +0.000013 PG 1701 PGCH  EXIT  INQUIRE_CHANNEL/EXCEPTION REASON(CHANNEL_NOT_FOUND) CONTAINER_POOL_TOKEN(00000000)
__       +0.000013 PG 1700 PGCH  ENTRY CREATE_CHANNEL CHANNEL_NAME(PRIME2) LINK_LEVEL(CURRENT) CURRENT_CHANNEL(NO)
__       +0.000014 SM 0301 SMGF  ENTRY GETMAIN SUBPOOL_TOKEN(486155B4 , 0000006C) GET_LENGTH(40) SUSPEND(YES) INITIAL_IMAGE(00) REM
__       +0.000016 SM 0302 SMGF  EXIT  GETMAIN/OK ADDRESS(17D881B0)
__       +0.000016 PG 1800 PGCP  ENTRY CREATE_CONTAINER_POOL CCSID(25) IMPORTED(NO) CHANNEL_RELATED(YES)
__       +0.000017 SM 0301 SMGF  ENTRY GETMAIN SUBPOOL_TOKEN(48615680 , 0000006D) SUSPEND(YES) INITIAL_IMAGE(00) REMARK(CPCB) LOCK_
__       +0.000017 SM 0302 SMGF  EXIT  GETMAIN/OK ADDRESS(17D89150)
__       +0.000017 PG 1801 PGCP  EXIT  CREATE_CONTAINER_POOL/OK POOL_TOKEN(17D89150)
__       +0.000017 PG 1701 PGCH  EXIT  CREATE_CHANNEL/OK CHANNEL_TOKEN(17D881B0) CONTAINER_POOL_TOKEN(17D89150)
__       +0.000018 PG 1900 PGCR  ENTRY PUT_CONTAINER POOL_TOKEN(17D89150) CONTAINER_NAME(HEADER) CALLER(EXEC) DATATYPE(CHAR) ITEM_D
__       +0.000020 AP 4800 CCNV  ENTRY VERIFY_IANA_CCSID IANA_CCSID(0037)
__       +0.000021 AP 4801 CCNV  EXIT  VERIFY_IANA_CCSID/EXCEPTION REASON(IANA_CCSID_NOT_KNOWN) IBM_CCSID(0)
__       +0.022637 AP 4800 CCNV  ENTRY VERIFY_CICS_CCSID CICS_CCSID(0037)
__       +0.022643 AP 4801 CCNV  EXIT  VERIFY_CICS_CCSID/OK IBM_CCSID(25) CLIENT_INDEX(0) SERVER_INDEX(1)
__       +0.022649 SM 4201 S2GF  ENTRY GETMAIN SUBPOOL_TOKEN(00000050_40804584 , 00000000_00000071) GET_LENGTH(1000) SUSPEND(YES) R
__       +0.022652 SM 4202 S2GF  EXIT  GETMAIN/OK ADDRESS(00000050_40D01000)
__       +0.022656 PG 1901 PGCR  EXIT  PUT_CONTAINER/OK CONTAINER_TOKEN_OUT(1798B730) GENERATION_NUMBER(1) INITIAL_GENERATION(1)
__       +0.022657 AP F802 EIBAM EXIT  PUT_CONTAINER RESP=0 RESP2=0
__       +0.022660 AP 00E1 EIP   EXIT  PUT-CONTAINER OK REQ(00F4)
__       +0.023157 AP 21E0 JCICS EXIT  Container_PUT
****************************************************************** Bottom of Data ******
```

JCICS PUT CONTAINER

A single JCICS request might issue multiple EXEC CICS calls

When the Java transaction is correlated to its AOR task

```
C\PROF Application Events                                                    Row 1 of 312 More: >
Command ===>                                                                 Scroll ===> CSR

Tran: MOBJ  Start: 2017-01-20 14:58:38.056913 Response: 2.559088   Task: 48766

/    Relative APPLID   Program     Elapsed Call            Resource  EIBRESP EIBRESP2 Command
__  +0.000000 CICSJOR  DFHSJTHP            ATTACH TASK                              ATTACH/OK TRANNUM(0048766C)
__  +0.024971 CICSJOR  DFHSJTHP   0.000119 JCICS                                    DTCTask_getCommonData
__  +0.025098 CICSJOR  DFHSJTHP   0.000010 JCICS                                    DTCTask_GETPROGRAM
__  +0.025330 CICSJOR  DFHSJTHP   0.000012 JCICS                                    Wrapper_GetCommArea envp(000000004C9B2100)
__  +0.034056 CICSJOR  DFHSJTHP   0.000012 JCICS                                    Task_GETCURRENTCHANNEL
__  +0.034087 CICSJOR  DFHSJTHP   0.023157 JCICS                                    Container_PUT
__  +0.057284 CICSJOR  DFHSJTHP   0.001345 JCICS                                    Container_PUT
__  +0.058655 CICSJOR  DFHSJTHP   0.000027 JCICS                                    Container_PUT
__  +0.065610 CICSJOR  DFHSJTHP            JCICS                                    DTCProgram_LINK envp(000000004C9B2100)
__  +0.065638 CICSJOR  DFHSJTHP   0.001540 LINK            EYU9XLOP  OK       0    LINK
__  +0.067205 CICSJOR  EYU9XLOP   0.001646 ASSIGN                      OK       0    ASSIGN OK
__  +0.068853 CICSJOR  EYU9XLOP   0.000010 HANDLE ABEND               OK       0    HANDLE-ABEND OK
__  +0.068948 CICSJOR  EYU9XLOP   0.010467 RETURN                                   SET_UOW/OK
__  +0.091191 CICSAOR1 DFHMIRS             ATTACH TASK                              ATTACH/OK TRANSACTION_TOKEN(17EBF300 , 0013525C) TRANNUM(0013525C)
__  +0.110682 CICSAOR1 DFHMIRS             START PROGRAM   DFHMIRS                  START_PROGRAM PROGRAM(DFHMIRS) CEDF_STATUS(CEDF) EXEC_SET(FULLAPI)
__  +0.114018 CICSAOR1 DFHMIRS   0.000036 LINK            MOBILE2   OK       0    LINK
__  +0.116491 CICSAOR1 MOBILE2   0.000005 ADDRESS                     OK       0    ADDRESS OK
__  +0.116499 CICSAOR1 MOBILE2   0.000017 GETMAIN                      OK       0    GETMAIN OK
__  +0.116521 CICSAOR1 MOBILE2   0.000000 ADDRESS                     OK       0    ADDRESS OK
__  +0.116523 CICSAOR1 MOBILE2   0.001259 GETMAIN                      OK       0    GETMAIN OK
__  +0.117888 CICSAOR1 MOBILE2   0.000006 GETMAIN                      OK       0    GETMAIN OK
__  +0.117920 CICSAOR1 MOBILE2   0.000001 ADDRESS                     OK       0    ADDRESS OK
__  +0.117923 CICSAOR1 MOBILE2   0.000001 ADDRESS                     OK       0    ADDRESS OK
__  +0.117924 CICSAOR1 MOBILE2   0.000004 ASSIGN                      OK       0    ASSIGN OK
__  +0.117987 CICSAOR1 MOBILE2   0.000012 GET CONTAINER               OK       0    GET-CONTAINER OK
__  +0.118004 CICSAOR1 MOBILE2   0.000000 ADDRESS                     OK       0    ADDRESS OK
__  +0.118008 CICSAOR1 MOBILE2   0.001657 QUERY SECURITY             OK       0    QUERY-SECURITY OK
__  +0.119816 CICSAOR1 MOBILE2   0.001525 SQL SELECT     ACCTDB2   0        0    APPLICATION SQLCODE 0 RETURNED ON EXEC SQL SELECT
__  +0.121355 CICSAOR1 MOBILE2   0.000006 GETMAIN                      OK       0    GETMAIN OK
__  +0.121364 CICSAOR1 MOBILE2   0.002363 LINK            ACC246    OK       0    LINK
__  +0.123985 CICSAOR1 ACC246    0.000012 GETMAIN                      OK       0    GETMAIN OK
__  +0.124010 CICSAOR1 ACC246    0.001190 ASSIGN                       OK       0    ASSIGN OK
__  +0.125201 CICSAOR1 ACC246    0.000011 LINK            CUS147    OK       0    LINK
__  +0.125272 CICSAOR1 CUS147    0.001570 GETMAIN                      OK       0    GETMAIN OK
__  +0.126863 CICSAOR1 CUS147    0.001737 SQL SELECT      ACCTDB2   0        0    APPLICATION SQLCODE 0 RETURNED ON EXEC SQL SELECT
```

Cross over into the AOR

EI level 1 is used in production to save MIPS

Rich EXEC CICS command formatting and program statement numbers are not available

Multiple program links

Where are the delays? Which program are they in?

```
C\PROF Application Events                                          Row 54 of 312 More: >
Command ===> _____________________________________________________ Scroll ===> CSR

Tran: MOBJ  Start: 2017-01-20 14:58:38.056913 Response: 2.559088   Task: 48766

/    Relative APPLID   Program     Elapsed Call          Resource  EIBRESP EIBRESP2 Command
__  +0.224741 CICSAOR1 FINA89    0.001455 GET CONTAINER           OK      0        GET-CONTAINER OK
__  +0.226199 CICSAOR1 FINA89    0.000008 GET CONTAINER           OK      0        GET-CONTAINER OK
__  +0.226210 CICSAOR1 FINA89    0.000001 ADDRESS                 OK      0        ADDRESS OK
__  +0.226219 CICSAOR1 FINA89    0.003741 SQL OPEN       ACCTDB2  0       0        APPLICATION SQLCODE 0 RETURNED ON EXEC SQL OPEN
__  +0.229972 CICSAOR1 FINA89    0.001501 SQL FETCH      ACCTDB2  +100    100      APPLICATION SQLCODE 100 RETURNED ON EXEC SQL FETCH
__  +0.231484 CICSAOR1 FINA89    0.002544 SQL CLOSE      ACCTDB2  0       0        APPLICATION SQLCODE 0 RETURNED ON EXEC SQL CLOSE
__  +0.234230 CICSAOR1 FINA89    0.002118 SQL DELETE     ACCTDB2  +100    100      APPLICATION SQLCODE 100 RETURNED ON EXEC SQL DELETE
__  +0.236358 CICSAOR1 FINA89    0.003571 SQL DELETE     ACCTDB2  +100    100      APPLICATION SQLCODE 100 RETURNED ON EXEC SQL DELETE
. . .
__  +2.353995 CICSAOR1 POR911    0.000606 SQL DELETE     ACCTDB2  +100    100      APPLICATION SQLCODE 100 RETURNED ON EXEC SQL DELETE
__  +2.398089 CICSAOR1 POR911    0.001746 GET CONTAINER           LENGERR 11       GET-CONTAINER LENGERR
__  +2.399836 CICSAOR1 POR911    0.000006 GET CONTAINER           OK      0        GET-CONTAINER OK
__  +2.399844 CICSAOR1 POR911    0.000002 ADDRESS                 OK      0        ADDRESS OK
__  +2.399847 CICSAOR1 POR911    0.000000 ADDRESS                 OK      0        ADDRESS OK
__  +2.399858 CICSAOR1 POR911    0.002066 SQL SELECT     ACCTDB2  0       0        APPLICATION SQLCODE 0 RETURNED ON EXEC SQL SELECT
__  +2.401935 CICSAOR1 POR911    0.001275 LINK           AR047V   OK      0        LINK
__  +2.403344 CICSAOR1 AR047V    0.000005 ADDRESS                 OK      0        ADDRESS OK
__  +2.403359 CICSAOR1 AR047V    0.001509 LINK           ACCT10   OK      0        LINK
__  +2.404921 CICSAOR1 ACCT10    0.000007 GETMAIN                 OK      0        GETMAIN OK
__  +2.404941 CICSAOR1 ACCT10    0.000001 ADDRESS                 OK      0        ADDRESS OK
__  +2.406640 CICSAOR1 AR047V    0.000007 ASSIGN                  OK      0        ASSIGN OK
__  +2.406647 CICSAOR1 AR047V    0.000007 WRITEQ TD               QIDERR  0        WRITEQ-TD QIDERR
```

Expensive DB2 calls with bad SQL codes: can they be avoided?

Response time so far to this point of transaction processing

EXEC CICS and SQL calls: elapsed time and response codes

EXEC CICS calls with (probably) unexpected response codes?
Ticking time bombs . . .

View trace events associated with the application call

```
C\PROF Trace Events                                                    Record 00000001 More: < >
Command ===> _____________________________________________________________ Scroll ===> CSR
   14:58:40.455003 Level 0  Gap < 4          >                          Format ===> SNGL
/  Time (Relative) ------------------------------------------------------------------------
__ 14:58:40.455003 AP 00E1 EIP    ENTRY GET-CONTAINER REQ(0004)
__       +0.000000 AP D500 UEH    EVENT LINK-TO-USER-EXIT-PROGRAM CMRXEIO3 AT EXIT POINT XEIIN
__       +0.000000 AP D501 UEH    EVENT RETURN-FROM-USER-EXIT-PROGRAM CMRXEIO3 WITH RETURN CODE 0
__       +0.000001 AP F801 EIBAM ENTRY GET_CONTAINER
__       +0.000001 PG 1700 PGCH   ENTRY INQUIRE_CHANNEL CHANNEL_NAME(FUNDI2)
__       +0.000003 PG 1701 PGCH   EXIT  INQUIRE_CHANNEL/OK CONTAINER_POOL_TOKEN(1A40C120)
__       +0.000003 PG 1900 PGCR   ENTRY GET_CONTAINER_INTO POOL_TOKEN(1A40C120) CONTAINER_NAME(HEADER) CALLER(E
__       +0.000004 PG 1901 PGCR   EXIT  GET_CONTAINER_INTO/EXCEPTION REASON(MORE_DATA) USERACCESS(ANY) DATATYPE
__       +0.001741 PG 1900 PGCR   ENTRY GET_CONTAINER_LENGTH POOL_TOKEN(1A40C120) CONTAINER_NAME(HEADER) CALLER
__       +0.001743 PG 1901 PGCR   EXIT  GET_CONTAINER_LENGTH/OK USERACCESS(ANY) DATATYPE(CHAR) DATA_LENGTH(7E1)
__       +0.001745 AP F802 EIBAM EXIT  GET_CONTAINER RESP=22 RESP2=11
__       +0.001746 AP 00E1 EIP    EXIT  GET-CONTAINER LENGERR REQ(00F4)
```

EXEC CICS GET CONTAINER

Drill down to the call trace events

Options control the level of trace that is active

Identify the cause of delays

Diagnose problems associated with the call

```
C\PROF Trace Events                                                    Record 00000001 More: < >
Command ===> _____________________________________________________________ Scroll ===> CSR
   14:58:39.060836 Level 0  Gap < 4          >                          Format ===> SNGL
/  Time (Relative) ------------------------------------------------------------------------
__ 14:58:39.060836 AP 3180 D2EX1 ENTRY APPLICATION REQUEST - EXEC SQL DELETE
__       +0.001408 AP 3250 D2D2  ENTRY DB2_API_CALL CSUB_TOKEN(19BFB350)
__       +0.752111 AP 3251 D2D2  EXIT  DB2_API_CALL/OK
__       +0.752120 AP 3181 D2EX1 EXIT  APPLICATION SQLCODE 100 RETURNED ON EXEC SQL DELETE
__       +0.752136 AP D500 UEH   EVENT LINK-TO-USER-EXIT-PROGRAM CMRXEIO3 AT EXIT POINT XRMIOUT
__       +0.752147 AP D501 UEH   EVENT RETURN-FROM-USER-EXIT-PROGRAM CMRXEIO3 WITH RETURN CODE 0
__       +0.752152 AP 2521 ERM   EXIT  PLI-APPLICATION-CALL-TO-TRUE(DSNCSQL)
__       +0.752163 AP 2520 ERM   ENTRY PLI-APPLICATION-CALL-TO-TRUE(DSNCSQL)
```

EXEC SQL DELETE

The AOR program ends with an implicit syncpoint that includes DB2…

```
C\PROF Application Events                                                    Row 266 of 312 More: >
Command ===> ____________________________________________________________  Scroll ===> CSR

Tran: MOBJ  Start: 2017-01-20 14:58:38.056913 Response: 2.559088   Task: 48766

/    Relative APPLID   Program     Elapsed Call             Resource EIBRESP EIBRESP2 Command
__  +2.409809 CICSAOR1 MOBILE2    0.001586 GET CONTAINER              OK       0        GET-CONTAINER OK
__  +2.411398 CICSAOR1 MOBILE2    0.000004 PUT CONTAINER              OK       0        PUT-CONTAINER OK
__  +2.411405 CICSAOR1 MOBILE2    0.000008 LINK            FS982X     OK       0        LINK
__  +2.413669 CICSAOR1 FS982X     0.012402 RETURN
__  +2.413723 CICSAOR1 MOBILE2    0.001750 FREEMAIN                   OK       0        FREEMAIN OK
__  +2.415477 CICSAOR1 MOBILE2    0.000005 FREEMAIN                   OK       0        FREEMAIN OK
__  +2.416969 CICSAOR1 MOBILE2    0.007641 SQL SYNCPOINT   ACCTDB2    RMI=4    0        SYNCPOINT-MANAGER REQUEST
__  +2.441926 CICSAOR1 DFHMIRS    0.004272 SQL TASK-MGR    ACCTDB2    RMI=0    0        TASK-MANAGER REQUEST
__  +2.446739 CICSAOR1 DFHSJTHP            RELEASE TASK                                  RELEASE_XM_CLIENT TERM_TYPE(NORMAL)
__  +2.453362 CICSJOR  DFHSJTHP  0.000024 JCICS                                        Container_GETLENGTH
__  +2.453395 CICSJOR  DFHSJTHP  0.000267 JCICS                                        Container_GET
__  +2.453704 CICSJOR  DFHSJTHP  0.000034 JCICS                                        Container_GETLENGTH
__  +2.453752 CICSJOR  DFHSJTHP  0.001958 JCICS                                        Container_GET
__  +2.455731 CICSJOR  DFHSJTHP  0.000019 JCICS                                        Container_GETLENGTH
__  +2.455762 CICSJOR  DFHSJTHP  0.002014 JCICS                                        Container_GET
__  +2.548949 CICSJOR  DFHSJTHP  0.000017 JCICS                                        DTC_Clean
__  +2.554923 CICSJOR  DFHSJTHP            START PROGRAM    EYU9XLOP                   START_PROGRAM PROGRAM(EYU9XLOP)
__  +2.554941 CICSJOR  DFHSJTHP  0.000008 ASSIGN                     OK       0        ASSIGN OK
__  +2.556630 CICSJOR  DFHSJTHP  0.000010 HANDLE ABEND              OK       0        HANDLE-ABEND OK
__  +2.556749 CICSJOR  DFHSJTHP  0.002331 RETURN                                      COMMIT_UOW/OK FAILED_LINK(00000000)
__  +2.559088 CICSJOR  DFHSJTHP            RELEASE TASK                                  RELEASE_XM_CLIENT TERM_TYPE(NORMAL)
```

…and then control is returned across the MRO link back to Java

Minimal impact on CICS: C\Prof feature summary

Profile

- Application perspective of the trace
- Near real-time transaction behavior
- Deep dive into application events
- Trace across multiple CICS regions (MRO)

Snap

- Back-in-time snapshot of the CICS internal trace (look back in time)
- Capture first occurrence of problem
- Writes to an auxiliary trace data set (readable by C\Prof and existing tools)

Record

- Record the trace until you say stop
- Writes to auxiliary trace data sets
- Features of CICS auxiliary trace but with additional power and control

Load

- Load any auxiliary trace data set into the C\Prof transaction profiler
- Merge multiple or load one at a time
- Use auxiliary trace data sets produced by C\Prof or the CICS auxiliary trace