CICS Test Drive

Feel the Power
Introduction

Preface

Why is Threadsafe such a Hot Topic?

- *It can save you CPU*
- *It can take advantage of multiple engines on today's processors*
- *It can increase throughput*

It can save you $$$$$

CustomerThreadsafe experiences

- **Customers are reporting savings between 2-15%**
- **Danske Bank**
  - Savings around **300 MIPS** during the peak processing period – directly attributed to reduced TCB switching
  - Now considering CICS Transaction Server Version 3.2 with additional threadsafe enhancements
    - WMQ
    - VSAM local File Control and RLS
    - TCP/IP Sockets
- **“Large financial services organization in the US“**
  - Estimated **$32M per year** in savings based on data center chargeback reduction
  - Public domain information, delivered at Impact 2008
  - Analysis done by 1 non-dedicated systems programmer 6-7 months without threadsafe specific tools
- **“Major US bank”**
  - Saved **700 MIPs** by making one major application threadsafe

CICS Threadsafe - Real CPU Savings

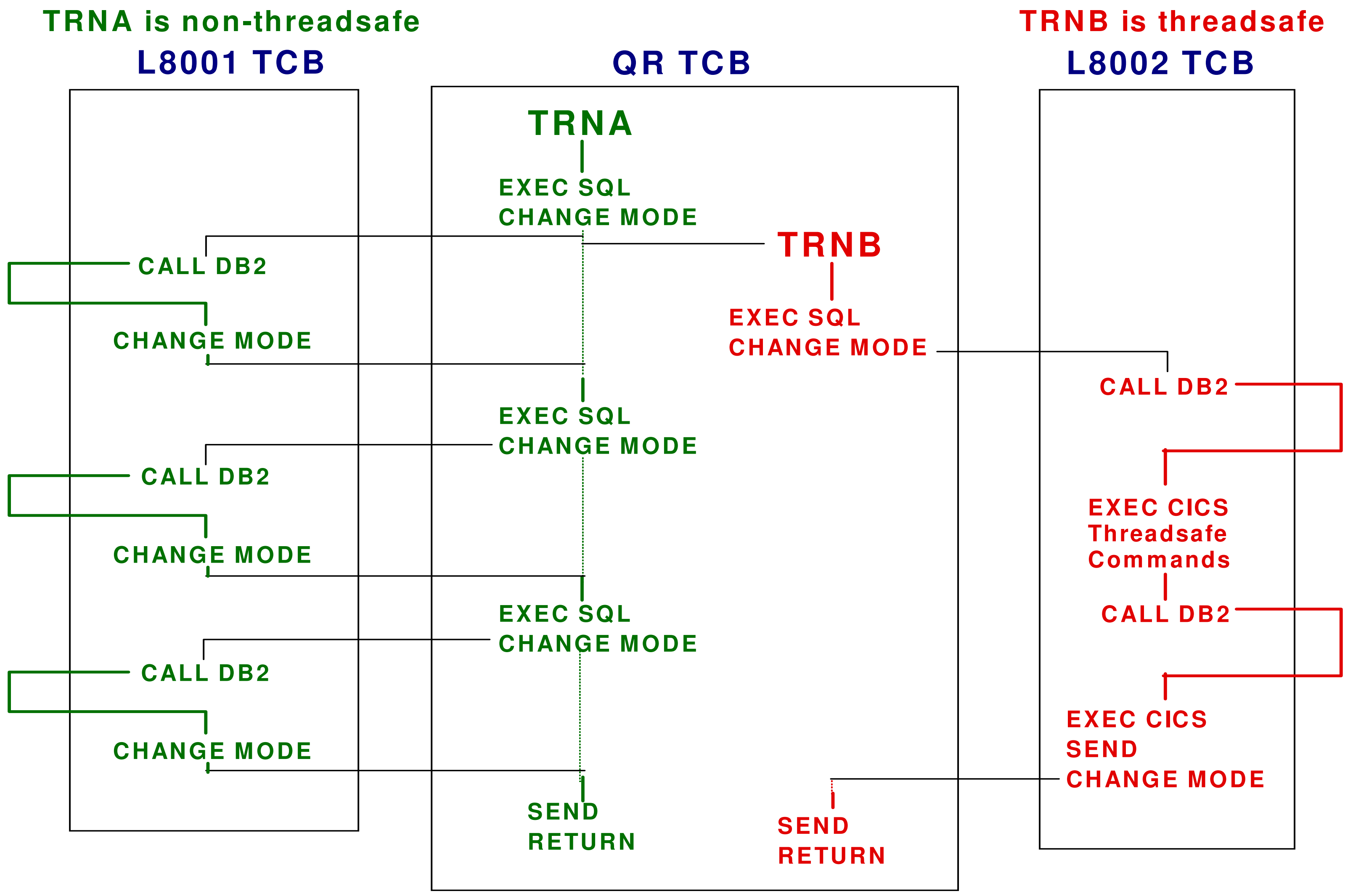
- Calculation for wasted CPU because of TCB switching
  - 2,000 instructions per TCB Switch * 6 cycles per instructions * .58 nanoseconds per Cycle = 6,960 for 1 switch
  - 1,000 TCB switches * 6,960 nanoseconds = .00696 seconds
  - Execute this program 1,000 times a day
    - TCB switching will waste 6.96 CPU seconds per day
  - Now execute this program 1 million times a day
    - **TCB switching will waste 116 CPU minutes per day**

  - Assumptions
    - Z9 processor is .58 nanoseconds per cycle
    - Average instruction requires 6 cycles
    - Each TCB switch is 2,000 instructions

Why make applications Threadsafe?

- Improve performance
  - CICS QR TCB is CPU constrained
  - Application tasks are waiting excessively for the QR TCB
  - The CICS region in general is CPU constrained
- Reduce cost
- Future positioning
  - OTE function introduced
  - TRUEs can exploit OTE
  - Full application use of open TCBs

CICS-DB2 Transactions in CICS TS 2.2 and higher



CICS Tools portfolio

CICS Performance Analyzer
- Comprehensive batch performance reporting and analysis for tuning and capacity planning

CICS Interdependency Analyzer
- Understand you active application inventory for efficient maintenance and upgrades

CICS Configuration Manager
- Manage, replicate, and deploy CICS system definitions

CICS VSAM Transparency
- Enable VSAM to DB2 migration without rewriting applications

CICS VSAM Recovery
- Automate recovery of lost VSAM data

IBM Session Manager
- Provide secure, reliable, and easy access to multiple z/OS and OS/390 applications from a single terminal

CICS Online Transmission Time Optimizer
- Optimize 3270 data streams to increase your system performance

CICS Batch Application Control
- Simplify and automate batch access to CICS resources

CICS Tools that can help – CICS PA

- Which TCBs did my transaction use?
  - How many TCB switches (change modes) occurred?
    - What was the Change Mode delay time?
  - How much Dispatch and CPU time did they use?
  - Performance Summary, List and List Extended Reports, …
  - Sample Report Forms …
    - CPU and TCB Usage, TCB Delays, Change Mode Delays, …
- Why did my transaction take so long?
  - Wait Analysis Report, Performance List Reports, …
- Which Transaction(s) used GETMAIN SHARED?
- Where did my transaction go?
  - Cross-System Report, …
  - Performance List, DB2 and WebSphere MQ Reports, …

CICS Tools that can help – CICS IA

- Are my programs using shared resources?
  - CWA
  - Global user exit global work areas
- Storage acquired explicitly by the application program with the GETMAIN SHARED option
  - Data only Load module
- Which non threadsafe CICS commands is my program using?
- Which TCB did my CICS commands run on?
- Program analysis with report on Threadsafe status

CICS Tools that can help – CICS CM

- How can I simplify and automate the management of my CICS threadsafe resources ?
  - Minimizes manual work by operators and system programmers
- How can I migrate the CICS threadsafe resources from different environments under a structured change control process?
  - Enables migration of CICS resources from different environments under a structured change control process
- Can I see CICS Threadsafe resource history as well as provides back-out to previous change level?
  - Provides complete audit history of all CICS resource modifications
  - Tracks resource history as well as provides back-out to previous change level
  - Provides detailed reports of CICS resources

CICS Test Drive

START YOUR ENGINES !

Terminology

- Quasi-reentrant (QR) TCB
  - The main CICS TCB under which all application code runs prior to OTE
  - CICS dispatcher sub-dispatches work, so each CICS task has a slice of the action
  - A CICS task gives up control via a CICS dispatcher wait
  - Only one CICS user task is active at any one time

- Quasi-reentrant programs
  - Same program can be invoked by more than one CICS task
  - But only one CICS task is active at any one time
  - Quasi-reentrancy allows programs to share virtual storage e.g. CWA without the need to protect against concurrent update
  - CICS code takes advantage of quasi-reentrancy, e.g. field CSACDTA in the CSA addresses the TCA of the currently dispatched CICS task.

Notes

- Prior to OTE, all application code runs under the main CICS TCB called the quasi-reentrant (QR) TCB. The CICS dispatcher subdispatches use of the TCB between the CICS tasks. Each task voluntarily gives up control when it issues a CICS service that issues a CICS dispatcher wait. There is only ever one CICS task active at any one time on the QR TCB.

- Programs are said to be quasi-reentrant programs because they take advantage of the behaviour of the CICS dispatcher and the QR TCB, in particular that there is only ever one CICS task active under the QR TCB. This means that although the same program can be being executed by multiple CICS tasks, only one of those CICS tasks is active at any one point in time. Contrast this with a situation whereby multiple instances of the same program are executing each under a separate TCB. In this situation multiple tasks would be active in the same program at the same time and the program would have to be fully MVS reentrant.

- Quasi reentrant programs can access shared resources such as the common work area (CWA) or shared storage obtained via EXEC CICS GETMAIN SHARED safe in the knowledge that they are the only CICS user task running at that instance.

- Field CSACDTA in the CICS CSA is a field that relies on quasi-reentrancy. It points to the TCA of the currently dispatched CICS task. It only has meaning when running under the QR TCB. In CICS TS 3.1 all CICS use of CSACDTA was removed and the field renamed CSAQRTCA. **In CICS TS 3.2 the field is loaded with a fetch protected address. If used, an abend ASRD will result.**

Terminology

- Threadsafe programs
  - Are capable of being invoked on multiple TCBs concurrently
  - Are normally read-only, they do not in general overwrite themselves
    - Could overwrite themselves if updates are serialized correctly like shared stg
  - Cannot rely on quasi-re-entrancy to serialise access to resources and storage
  - Must use serialisation techniques such as compare and swap (CS) or enqueue/dequeue to access shared resources with integrity
  - All programs accessing a shared resource must be made threadsafe e.g. an existing program's reliance on quasi-re-entrancy to serialise access to the CWA is made invalid if just one other program can run concurrently on another TCB and access the same CWA field
  - Are sometimes referred to as fully MVS reentrant programs
    - MVS reentrant is often misunderstood to mean that programs do not overwrite themselves. We use the term threadsafe to avoid confusion

Notes

- The term THREADSAFE is used in preference to the term 'fully MVS reentrant' as the latter term is often misunderstood to just mean that a program is linkedited with the RENT option and hence is read-only and does not overwrite itself.
- For a program to be threadsafe it has to behave correctly when invoked concurrently under multiple execution units, i.e. TCBs. If it accesses shared storage for example it must ensure that updates to shared storage are serialised. If the program overwrites itself then this is a kind of shared storage and has to be serialised accordingly.

Terminology

- CICSAPI - services available today under QR TCB
  - CICS command level application programming interface
  - CICS system programming interface
  - CICS Resource Manager Interface (RMI)
  - CICS Exit Programming Interface (XPI) - for Global User exits
  - Systems Application Architecture (SAA) Common Programming Interfaces
    - CPI-C and CPI-RR
  - LE callable services

- OPENAPI - additional APIs possible under Open TCBs
  - Use of MVS services
  - Use of a specified set of POSIX services via MVS Unix System Services

Notes

- CICSAPI is the term used to describe the set of services available today prior to OTE.
- OPENAPI is the term used to describe the set of apis available to a program once it is running on an open TCB. An openapi program can issue any of the services available for a cicsapi program but in addition it is able to issue 'foreign api' calls, like MVS services.

OTE enhancements in CICS TS 3.2: File Control – API and SPI

- File Control API Commands - READ, WRITE, REWRITE, DELETE, UNLOCK, STARTBR, RESETBR, READNEXT, READPREV and ENDBR
  - These commands are threadsafe when accessing Local VSAM and VSAM RLS files
    - ie can be executed on an open TCB or QR TCB
    - Local VSAM to be enabled by CICS TS 3.2 apar PK45345 and will require vsam apar OA23252
  - These commands are not be threadsafe when accessing for Coupling Facility Data tables, shared data tables, remote files or BDAM files
    - ie must be executed on QR TCB and CICS will force a TCB switch if necessary
- File Control SPI Commands
  - INQUIRE FILE is now threadsafe
  - SET FILE, INQUIRE DSNAME, SET DSNAME, DISCARD FILE, CREATE FILE are still non threadsafe

Notes

- In CICS TS 3.2 the File control API commands have been made threadsafe for a subset of the File Control function. It is gated upon the access method involved.
- All File control API commands that are issued against local VSAM files or VSAM RLS files are threadsafe, and so will execute on an open TCB without incurring a TCB switch back to QR TCB.
- All File control API commands that are issued against remote files, coupling facility data tables, shared data tables, and BDAM files are still non threadsafe, and so a TCB switch back to QR TCB will be incurred if the application is running on an open TCB at the time it issues the request.
- All SPI commands against all types of file are still non threadsafe, except for INQUIRE FILE which has been made threadsafe.

Notes

The CICS QR TCB is CPU constrained The CICS QR TCB is consistently reaching system CP SHARE(QR TCB is running at 100% CPU) and has to wait to be dispatched by the operating system. Every task in the Every task running under the QR TCB is being delayed. Defining transactions as threadsafe, processing as many tasks as possible on the L8 TCBs will remove this constraint on the QR TCB, and reduce the response times of both threadsafe and non-threadsafe transactions.

Application tasks are waiting excessively for the QR TCB The QR TCB is not CPU constrained, but application tasks are contending for their share of QR. Again, defining transactions as threadsafe and moving as many tasks as possible to L8 TCBs will reduce contention for QR, and reduce the response times of both threadsafe and non-threadsafe transactions.

The CICS region in general is CPU constrained The system as a whole is at or approaching 100% busy, and CICS is being constrained along with everything else.

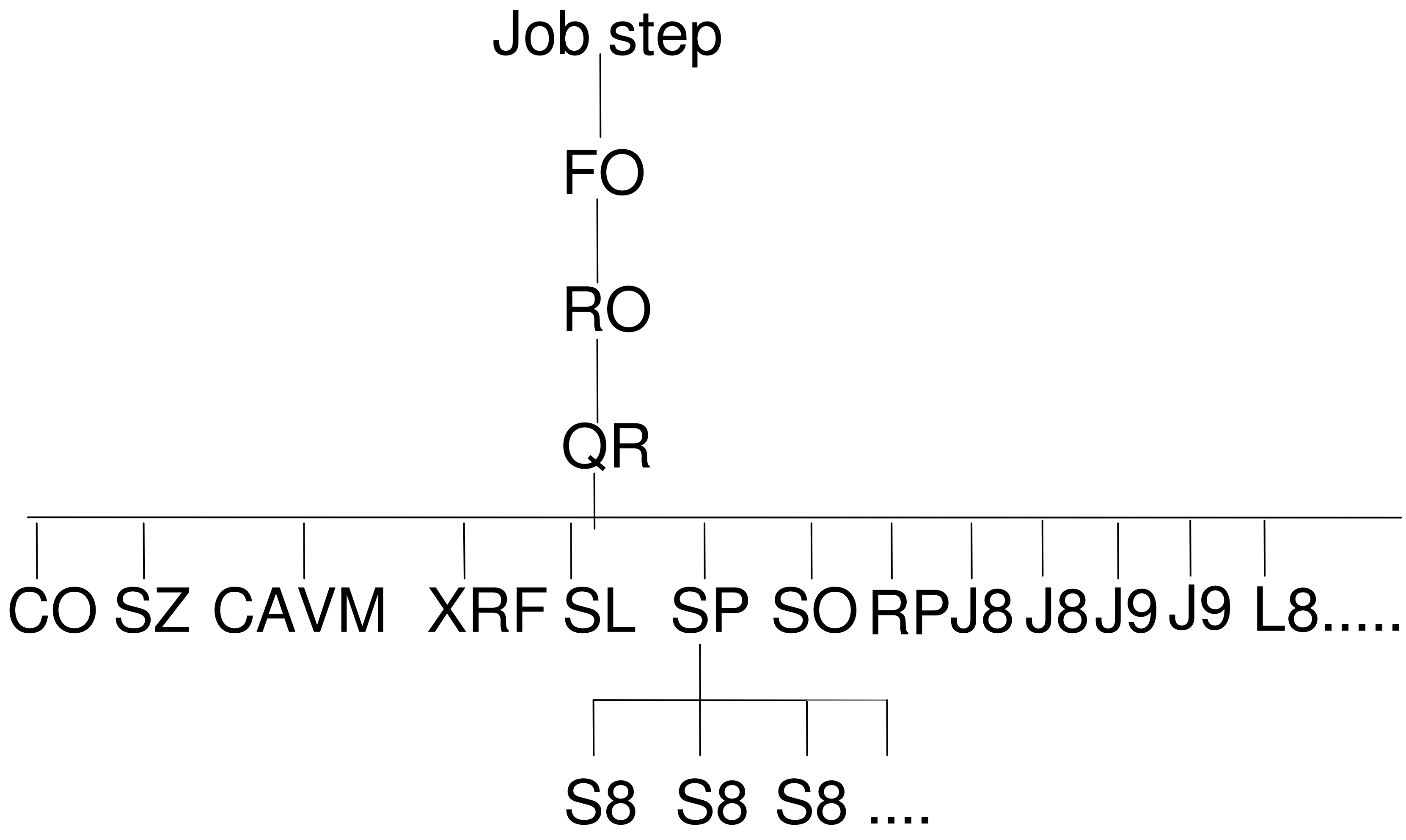
Depending on how an application is designed, defining it as threadsafe can significantly reduce the path length of application tasks. The transactions which will achieve the greatest CPU reduction will have the following characteristics:

- significant number of EXEC SQL calls invoked per task
- all programs invoked between the first and last EXEC SQL call in each task are defined as threadsafe
- all exits invoked as part of an EXEC SQL call are defined as threadsafe, and only contain threadsafe EXEC CICS commands
- all exits invoked between the first and last EXEC SQL call in each task are defined as threadsafe
- all EXEC CICS statements invoked between the first and last EXEC SQL call in each task are threadsafe Defining transactions with the above characteristics as threadsafe will all but eliminate TCB switches for the associated CICS tasks.

Notes

- The picture shows how the CICS-DB2 Attachment facility has been upgraded in CICS TS 2.2 to take advantage of OTE and use L8 open TCBs to call DB2 (when using DB2 V6 or higher) rather than privately managed DB2 Attach TCBs.
- TRNA executes a non threadsafe application program. Here the amount of TCB switching is the same as the previous release. Instead of switching to a DB2 Attach TCB, we use an L8 TCB. Switching is achieved via use of a disdpatcher domain Change_mode request, instead of use of WAIT/POST logic as in previous releases.
- TRNB executes an application that is threadsafe, and has been defined to CICS with CONCURRENCY(THREADSAFE). When executing the first EXEC SQL request, the TRUE is invoked on an open TCB, and that TCB is used to call DB2. On return from the RMI because the application is defined as threadsafe we stay on the open TCB and return to the application on it. Any threadsafe CICS commands issued by the application will be executed on the L8 TCB, as will any further SQL requests. For the EXEC CICS SEND command (which is non threadsafe) we switch back to the QR TCB. Control will stay on the QR TCB until another DB2 request is issued. In this example there are no more DB2 requests. The EXEC CICS RETURN is executed on the QR TCB. Not shown in the diagram is the end of task syncpoint which will result in four TCB switches for the two phase commit protocol, ie switch to L8 to issue PREPARE to DB2, then back to QR, then switch to L8 to issue COMMIT to DB2, then back to QR.

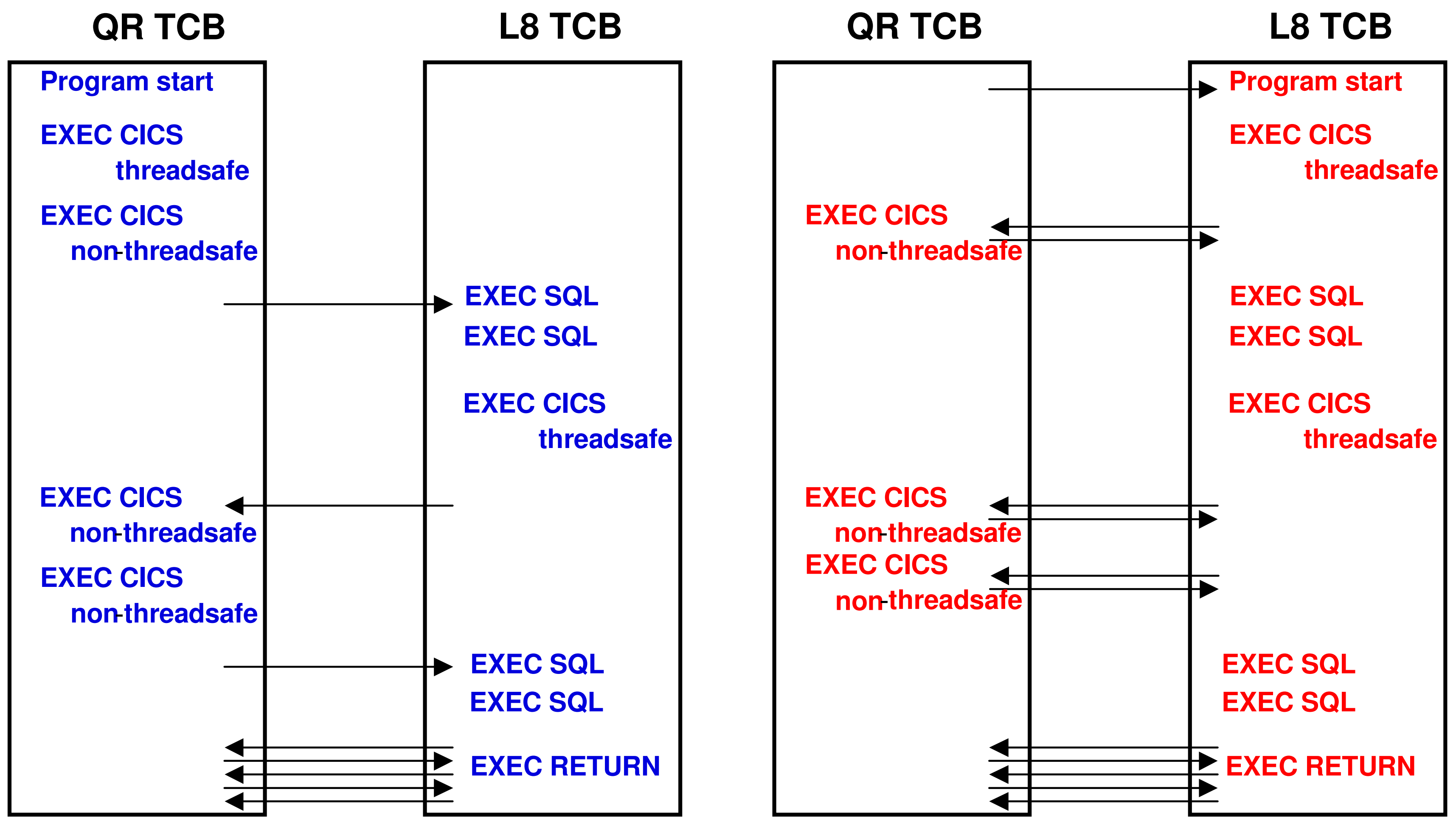
CICS TS 3.2 TCB hierarchy



Notes

- The picture shows the overall TCB hierarchy for CICS TS 3.2
- Under the QR TCB a number of daughter TCBs can be seen. Multiple J8 and J9 TCBs can be seen and represent multiple CICS tasks, each using a JVM. Also multiple L8 TCBs used by the CICS-DB2 Attachment Facility.
- Other than the L8, J8 and J9 TCBs, the other TCBs are all CICS system TCBs
  - FO is the file owning TCB used to offload various file operations
  - RO is the resource owning TCB under which program loads are performed
  - QR is the main CICS TCB under which applications are subdispatched
  - CO is the concurrent TCB used to offload I/O by various domains
  - SZ is used for FEPI
  - RP is used for ONC RPC calls
  - CAVM and XRF are used for XRF support
  - SL is the sockets domain listener TCB and SO is used by the Sockets domain
  - SO TCB owns the S8
  - Multiple S8 TCBs are used by Sockets domain for SSL support

Comparing threadsafe versus openapi - TCB switching



The program for transaction BLUE is defined THREADSAFE , API= CICSAPI

The program for transaction RED is defined THREADSAFE, OPENAPI, EXECKEY=CICS

Notes

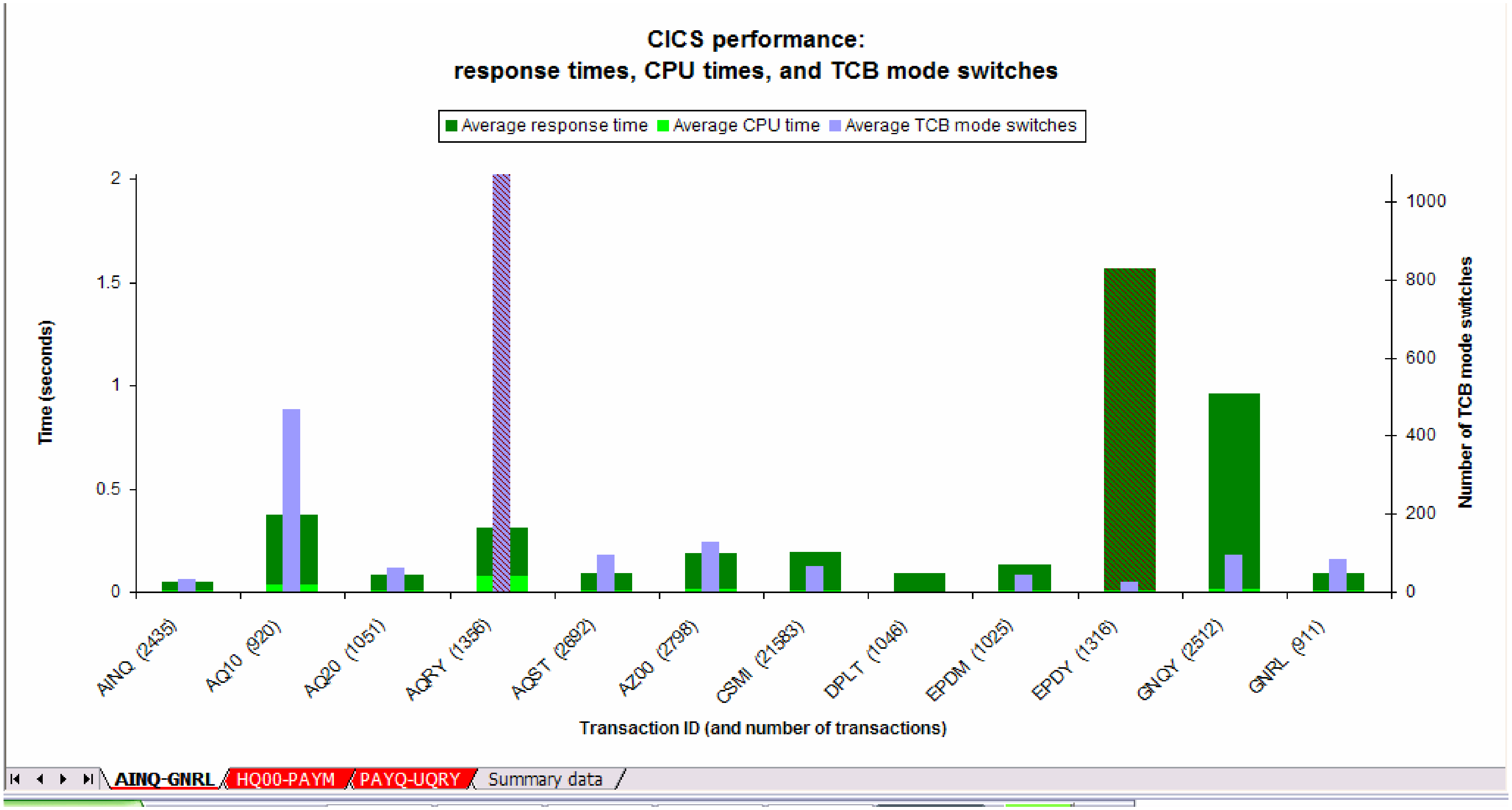
- In CICS Transaction Server R3.1, a program may be defined with the OPENAPI keyword, to indicate it should execute on an OPEN TCB (L8 or an L9, based on the EXECKEY specification). However, it should be noted *all non-threadsafe commands will continue to be processed on the QR TCB.*

- On the prior page notice the program used for transaction BLUE is defined as THREADSAFE, with API=CICSAPI. The execution of the task and TCB switching is the same as CICS Transaction Server R2.2 and R2.3. The program executes on the QR TCB until it issues an DB2 request, at which point it will switch to an L8 TCB. It will remain on the L8 until a non-threadsafe command is issued, causing it to switch back to the QR where it will remain until an other SQL request is issued.

- The application used for transaction RED is defined THREADSAFE, with API=OPENAPI and EXECKEY=CICS. The program is given control on an L8 TCB. Each time it issues a non-threadsafe command it will switch to the QR TCB to process the command and then return control to the application on the L8 TCB. SQL commands are executed on the L8 TCB along with any threadsafe commands. Notice, if there are many non-threadsafe commands, the overhead of switching can be greater than running with API=CICSAPI.

- Another important point to consider. If the program for transaction RED had been defined to run in EXECKEY USER, it will be given control on an L9 TCB, rather than an L8. The processing will be the same as noted above, EXCEPT when an SQL call is issued, the task is switched to an L8 TCB. Upon completion of the SQL request, control will be returned to the application on the L9 TCB.

Support Pac TCB Threadsafe Graph



This slide left blank intentionally