



Découverte des Mystères de l'attachement CICS/DB2

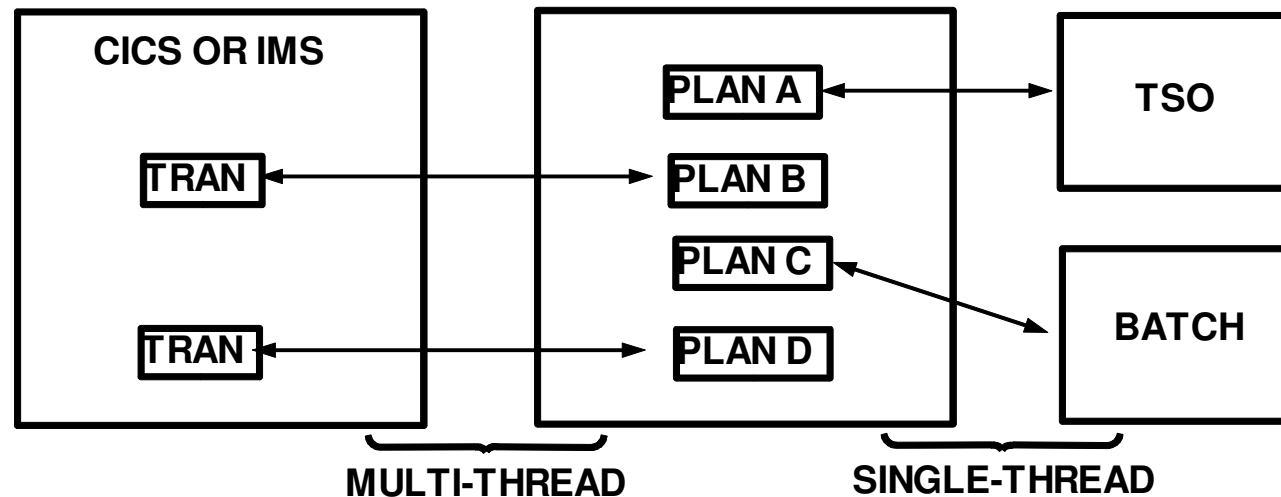
John Tilling
CICS Technical Planning & Strategy
IBM UK Laboratories

Tilling@uk.ibm.com

Agenda

- Overview of DB2 Terms and CICS-DB2 Attach architecture
- CICS-DB2 definitions: DB2CONN, DB2ENTRY, DB2TRAN
- Understanding TCBs
- Understanding threads
- Thread and signon reuse
- Reduction of TCB switching
- DB2 Group Attachment Feature
- Restart Lite support
- Summary

DB2 Connections



- Connection
 - Address Space TCB identifies itself to DB2 for Services
- Thread
 - A Bi-directional path between a user TCB and DB2 resources
- Plan
 - SQL Access Strategy used by the thread

Notes: DB2 Connections

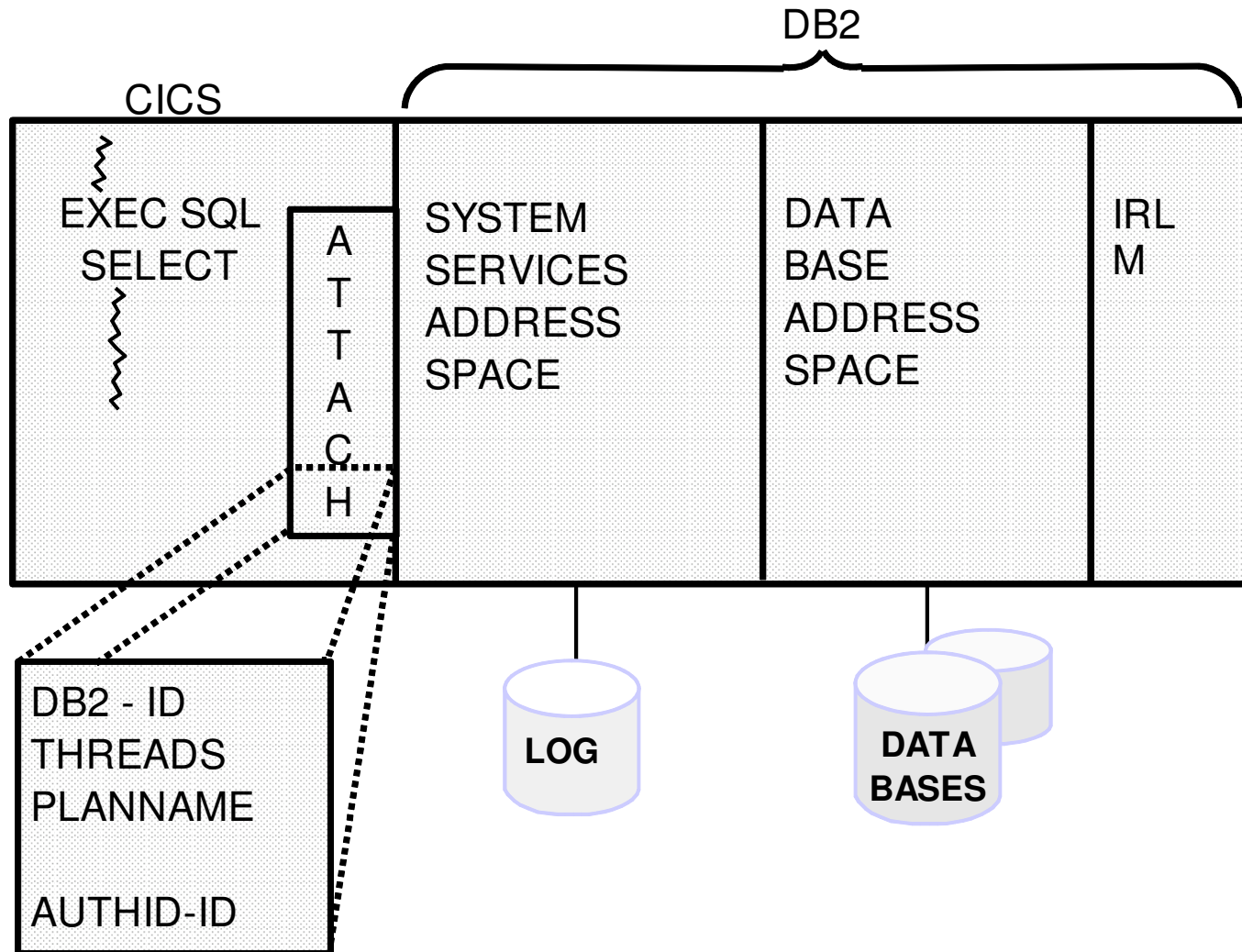
DB2 supports two types of connections:

- Single Thread Connection - Only one thread is allocated per connection, e.g.; TSO, batch.
- Multi-Thread Connection - Multiple threads can be established between a connected subsystem and DB2, e.g.; CICS, IMS.

Threads:

- Command Threads DO NOT have an associated plan.
- The Correlation ID is used to uniquely identify a thread.

Address spaces



Notes: Address spaces

- System Services (Control) Address Space provides connection support, DB2 logging, Accounting, Statistics traces.
- Database Services Address Spaces provides buffer management and database support.
- IRLM - Internal Resource Lock Manager works with DB2 to serialize access to DB2 data.
- DB2-ID: DB2's four char. system id.
- Thread: A bi-directional path between CICS and DB2 resources.
- Planname: A DB2 object that contains the SQL access strategy used by a thread.

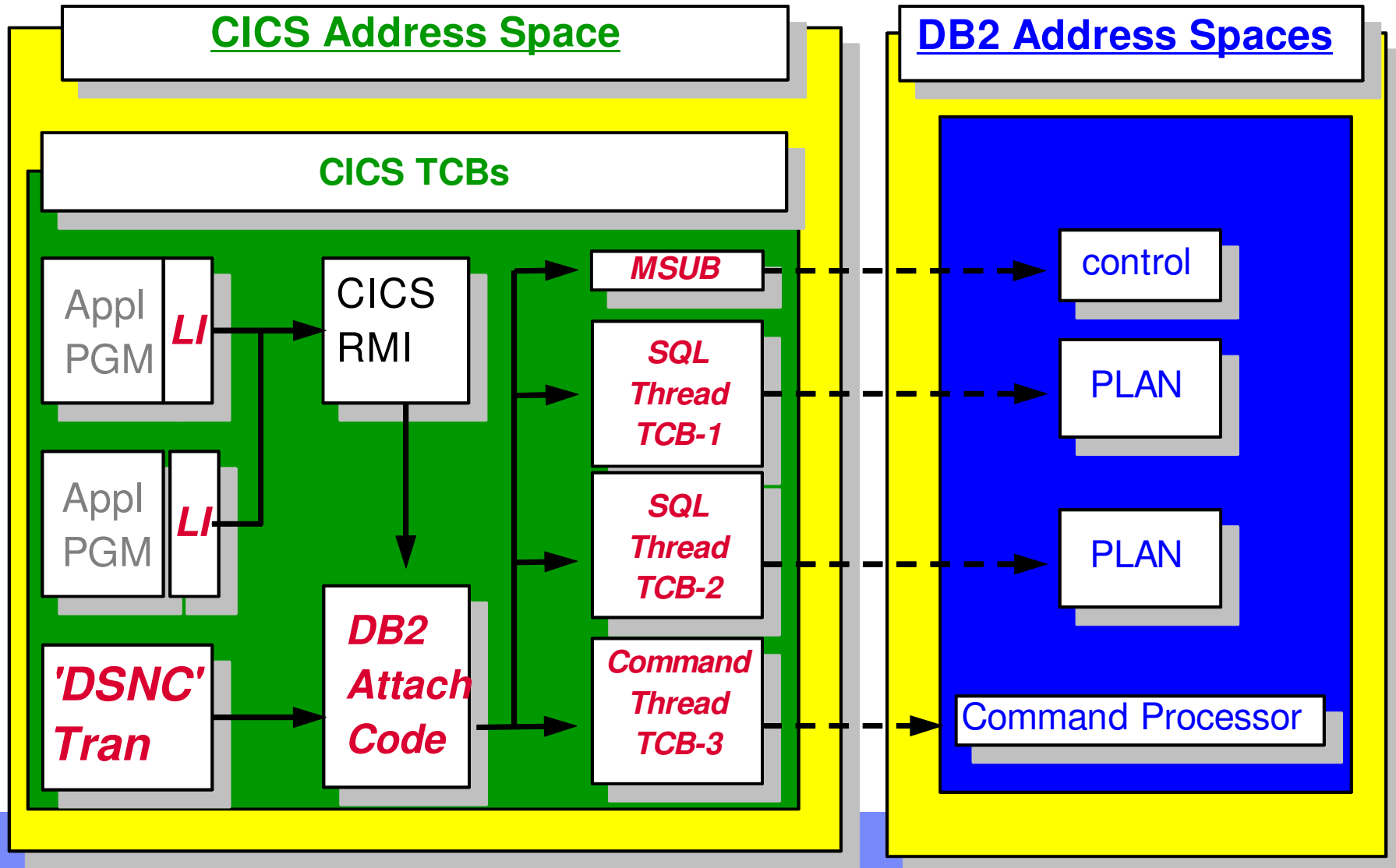
CICS DB2 Attachment Facility provides

- Application program Interface
 - EXEC SQL statements and Call level IFI requests
- Attachment commands
 - DSNB and CEMT
- CICS DB2 connection characteristics are defined by RDO
 - Number and types of threads
 - Planname selection criteria
 - Security
- CICS connects to a SINGLE DB2 Subsystem.
- CICS DB2 statistics and problem determination support
 - DSNB DISPLAY STAT, DFH0STAT and DFHSTUP
 - EDF, Messages, Abend Codes, standard CICS Dump support
- CICS TS supports all DB2 releases currently in service

Notes: CICS DB2 Attachment Facility provides

- The CICS DB2 attachment facility is provided with CICS TS. The CICS DB2 attachment facility provides CICS applications with access to DB2 data while operating in the CICS environment. CICS applications, therefore, can access both DB2 and CICS data.
- Attachment commands display and control the status of the facility.
- Resource Definition Online (RDO) is used to define the connection characteristics.
- You can name an individual DB2 subsystem or DB2 data sharing group. DB2 data sharing group support will be discussed later.
- There are no DB2 release dependencies within the attachment facility. It can connect to a DB2 subsystem running any supported level of DB2.

Looking inside



Notes: Looking inside

The CICS DB2 attachment and the associated CPU overhead exists mostly in the CICS address space.

The attachment processing to create and destroy resources it uses is done under the CICS TCBs.

The main DB2 subtask in the CICS address space is the master subtask (MSUB). This task coordinates DB2 start and stop notification, indoubt resolution and some housekeeping. In CICS TS 2.2 this job is done under the CICS RO TCB instead of MSUB. In CICS TS 2.2 MSUB is only present if connected to DB2 V5 or lower.

Most SQL processing occurs in thread TCBs. To a lesser extent, there is some processing in the DB2 address space. Prior to CICS TS 2.2, the thread TCBs are attached by MSUB and managed by the CICS-DB2 Attach. With CICS TS 2.2 when using DB2 V6 or higher the thread TCBs are CICS L8 open TCBs managed by the CICS Kernel and Dispatcher.

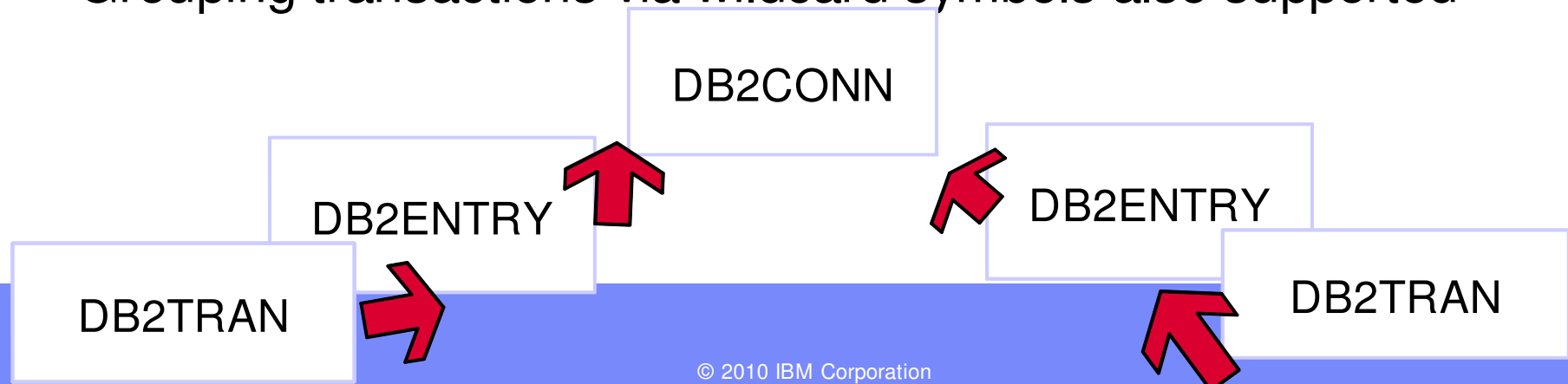
The attachment initiates the request in what is called the allied address space (the CICS region in this case).

The DSNB transaction is a special case which can pass DB2 commands (as opposed to SQL statements) to the connected DB2 subsystem.

All commands are supported except -START DB2

CICS-DB2 definitions

- DB2CONN
 - Global attributes of connection to a named DB2 system
 - POOL and COMMAND thread attributes
 - Only one can be installed at one time
- DB2ENTRY
 - Transaction Id with thread and plan selection requirements
 - Grouping transactions via wildcard symbols (* and +)
 - Installed after DB2CONN
- DB2TRAN
 - Additional transaction Id relating to DB2ENTRY
 - Grouping transactions via wildcard symbols also supported



Notes: CICS- DB2 Definitions

- The DB2CONN, DB2ENTRY and DB2TRAN resource definition types are added to the list of resources you can define in the CICS CSD.
- The DB2CONN resource definition type specifies the global attributes of the connection between CICS and a named DB2 subsystem.
- The DB2ENTRY defines resources used by specific CICS transactions or groups of transactions using generic resource names (wildcard values are '*' and '+')
 - An asterisk(*) can be added to a transaction name to produce the effect on any value making it wild. A plus sign (+) is allowed in any position to represent any single character.
- The DB2TRAN resource definition allows additional transactions to be associated with a particular DB2ENTRY.
- You can install a DB2ENTRY only if you have previously installed a DB2CONN. New DB2ENTRYs can be installed at any time, even when the CICS adapter is connected to DB2.
- If you discard a DB2ENTRY, you will make corresponding DB2TRAN an 'orphan'. If you then run a transaction, a message is sent to the CDB2 transient data destination, and the request is rerouted to the pool.

DB2CONN

```

DEFINE DB2CONN(RCTA1)
OVERTYPE TO MODIFY
CEDA DEFine DB2Conn( RCTA1 )
DB2Conn ==> RCTA1
Group ==>
DEscription ==>

CONNECTION ATTRIBUTES
Connecterror ==> Sqlcode      Sqlcode | Abend
Db2id      ==> ___
MSGQUEUE1  ==> CDB2
MSGQUEUE2  ==>
MSGQUEUE3  ==>
Nontermrel ==> Yes           Yes | No
Purgecycle ==> 00 , 30      0-59
Sgnid      ==>
STANdbymode ==> Reconnect    Reconnect | Connect | Noconnect
STATsqueue ==> CDB2
TCblimit   ==> 0012         4-2000
THREADError ==> N906D      N906D | N906 | Abend

```

Notes: Define Panel for DB2CONN(1)

- DB2CONN - Name of connection definition and can be up to eight characters in length.
- DB2ID (DSN | blank | name) - Four Character DB2 system name.
- PURGECYCLE (0 | value 1, 30 | value 2) - The duration in minutes and seconds, of the purge cycle for protected threads. The default is 30 seconds. Protected thread will be covered later in this session.
- SIGNID (applid | value) - The authorization ID to be used by this CICS system. The default is blanks which are replaced by the APPLID of the CICS system.
- STANDBYMODE - Specifies the action to be taken by the attachment if DB2 is not active and a connection attempt is made.
 - CONNECT - Attachment will wait for DB2 to be active.
 - NOCONNECT - Attachment requires a manual start.
 - RECONNECT - Same as CONNECT, except if DB2 should terminate and restart, the attachment will reconnect.
- TCBLIMIT(12 | value) specifies the maximum number of TCBs that can be used to process DB2 requests

DB2CONN (continued)

POOL THREAD ATTRIBUTES

ACCountrec ==>	None	None TXid <u>TAsk</u> Uow
AUTHId ==>		
<u>AUTHType</u> ==>	Userid	Userid Opid Group Sign TErm
	TX	
DROLLBACK ==>	Yes	Yes No
PLAN ==>		
PLANEXITName ==>		
PRiority ==>	High	High Equal Low
THREADLimit ==>	0003	3-2000
THREADWait ==>	Yes	Yes No

COMMAND THREAD ATTRIBUTES

COMAUTHId ==>		
COMAUTHType ==>	Userid	Userid Opid Group Sign TErm
	TX	
COMTHreadlim ==>	0001	0-2000

Notes: Define Panel for DB2CONN(2)

Pool Thread Attributes

- PLAN (name) specifies the name of the plan to be used for all pool threads. If PLAN is specified, PLANEXITNAME may not be coded.
- PRIORITY (HIGH | EQUAL | LOW) specifies the MVS dispatching priority of the pool thread subtask related to the CICS main task (QR TCB).
- THREADLIMIT (3 | 2000) specifies the current maximum number of pool threads that the CICS DB2 attachment allows to be active before requests are made to wait or are rejected (subject to the THREADWAIT attribute).
- THREADWAIT (YES | NO) specifies whether or not the transaction should wait for a pool thread, or be abended (Abend Code AD3T) if the number of active pool threads reaches the thread limit.

Command Thread Attributes

- COMTHREADLIM (1 | 2000) specifies the maximum number of command threads..

DB2ENTRY

```

DEFINE DB2ENTRY(ENTRY1)
OVERTYPE TO MODIFY
CEDA DEFine DB2Entry( ENTRY1 )
DB2Entry ==> ENTRY1
Group ==>
DEscription ==>

THREAD SELECTION ATTRIBUTES
TRansid ==>

THREAD OPERATION ATTRIBUTES
ACcountrec ==> None None | TXid | TAsk | Uow
AUTHId ==>
AUHTYpe ==> Userid Userid | Opid | Group | Sign | TErm
| TX
DROLLBACK ==> Yes Yes | No
PLAN ==>
PLANEXITName ==>
PRiority ==> High High | Equal | Low
PROtectnum ==> 0000 0-2000
THREADLimit ==> 0000 0-2000
THREADWait ==> Pool Pool | Yes | No

```

Notes: Define Panel for DB2ENTRY

- DB2ENTRY(Name) - One to eight character name to identify this DB2 entry definition.
- TRANSID - The transaction identifier associated with this DB2 entry. Only one transaction can be specified here. The use of one or more wildcard characters in the TRANID allows a group of transactions to be represented.
- PLAN - Specifies the Plan name assigned to this definition.
- PLANEXITNAME - Dynamic plan exit user program(DSNCUEXT)
- PROTECTNUM (Q | value) specifies the maximum number of protected threads allowed for this DB2 entry definition.
- THREADLIMIT (Q | 2000) - Maximum number of threads for this transaction group.
- THREADWAIT(POOL | YES | NO) - Specifies whether a transaction should wait (YES) for an entry thread, be abended (NO) or directed to the POOL, If the transaction group threads are in use.

DB2TRAN

```
DEFINE DB2TRAN(TXID1)
OVERTYPE TO MODIFY
CEDA DEFine DB2Tran( TXID1 )
DB2Tran ==> TXID1
Group ==>
DEscription ==>
Entry ==>
Transid ==>
```

Notes: Define Panel for DB2TRAN

- DB2TRAN - Name of definition.
- ENTRY - Name of DB2ENTRY in which this transaction is assigned.
- TRANSID - Name of assigned transaction or wildcard tranid.

Understanding TCBs

- Prior to OTE support in CICS TS 2.2, the CICS-DB2 Attach created/destroyed TCBs used to access DB2
 - TCBLIMIT on the DB2CONN controlled how many TCBs
 - A thread was built and was permanently associated with a TCB
- From CICS TS 2.2 onwards, CICS open (L8) TCBs are used
 - Number of open TCBs governed by MAXOPENTCBS in the SIT
 - TCBLIMIT (really now it should be called CONNECTIONLIMIT limit) is a subset of MAXOPENTCBS
 - How many connections into DB2
 - Threads are plugged/unplugged from L8 TCBs

Understanding TCBs

- MAXOPENTCBS needs to be at least as large TCBLIMIT on DB2CONN
 - You cannot use a connection into DB2 without a TCB
- MAXOPENTCBS will normally be greater than TCBLIMIT
 - To cater for other use of L8/L9 TCBs eg MQ, sockets, CICS Webservice pipelines
- If you use transaction isolation in production set MAAXOPNTCBS to your MXT value
 - Avoids contention for TCBs in subspaces versus basepsace
- CICS manages the open TCB pool and detaches TCBs that have not been used recently (30 minutes)

Understanding threads

- A thread is required to execute a DB2 request
 - A thread is associated with a DB2 Plan
 - A thread requires a DB2 connection
 - A connection requires a TCB
- As well as limiting Connections, you can limit the number of threads in the pool and in individual DB2ENTRYs (THREADLIMIT)
 - The sum of all THREADLIMITs should not exceed TCBLIMIT
 - You cannot use a thread without a DB2 connection

Getting a thread

A task with SQL requests can

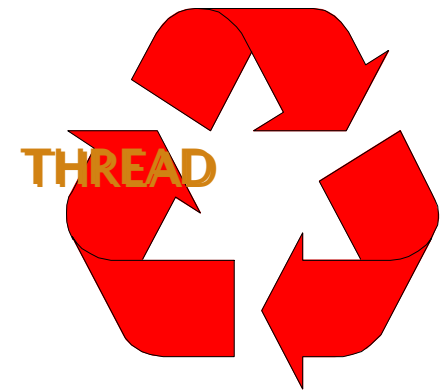
reuse an existing thread that another task used previously or allocate a new thread.

In order to reuse a thread

1. both tasks must be on the same DB2ENTRY or in the pool
2. both tasks must use the same plan.

A thread is released for reuse

- a. after SYNCPOINTS for terminal driven transactions (& non terminal transactions if NONTERMREL(YES)) that do not have open cursors using the HOLD option and special registers are in the initial state and
- b. at EOT for all transactions.



Notes: getting a thread

The most efficient way for a transaction to execute an SQL statement is to reuse a thread created by a previous transaction rather than allocating a new thread.

A thread can be reused only if the DB2ENTRY associated with the transaction is the same as the previous transaction and the plan is the same as the previous transaction.

A thread becomes available for reuse after SYNCPOINTs for terminal driven transactions if the thread is in the initial state. Initial state means that all the modifiable special registers are at their initial value and there are no held cursors.

For non terminal driven transactions prior to CICS TS 1.2, threads are only released at end of task. With CICS TS 1.2 you can set option NONTERMREL(YES) on the DB2CONN definition to cause non-terminal driven transactions to release their thread at syncpoint, subject to the same rules above.

The modifiable special registers include:

Register	Earliest DB2	Comments
CURRENT PACKAGESET	DB2 V2	
CURRENT SQLID	DB2 V2	
CURRENT DEGREE	DB2 V3	Parallel Queries
CURRENT SERVER	DB2 V3	Connect type-2
CURRENT RULES	DB2 V4	Standards
CURRENT LOCALE LC_CTYPE	DB2 V6	Locale
CURRENT OPTIMIZATION HINT	DB2 V6	
CURRENT PATH	DB2 V6	
CURRENT PRECISION	DB2 V6	

To Reuse or not reuse

When is thread reuse desirable?

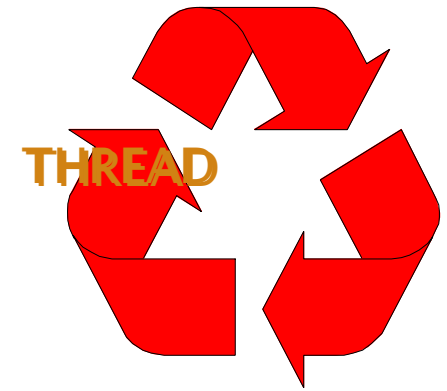
Primarily short transactions

CPU savings less significant for long transactions

When can thread reuse occur?

THREADWAIT=YES

or PROTECTNUM>0



DB2ENTRY screen fragment

```

CEDA  ALter DB2Entry( NO          )
DB2Entry      : WRNO
Group         : RCTA
DEscription  ==> New orders
THREAD SELECTION ATTRIBUTES
TRansid       ==> NO*
PLAN          ==> CRWWPPNO
PLANExitname ==>
PROtectnum    ==> 0001          0-2000
THREADLimit   ==> 0001          0-2000
THREADWait    ==> Yes          Pool | Yes | No
  
```

Notes: To reuse or not reuse

Generally, thread reuse is the desirable choice for CICS transactions using DB2. Typical transactions are:
short
high volume

Thus, the CPU savings of avoiding the creation of a new thread is significant.

To reuse a thread the DB2 thread definition (DB2ENTRY or DB2CONN for pool entries) must have either
THREADWAIT=YES to allow the transaction to wait for a thread already in use or
PROTECTNUM>0 to allow unused threads to wait for a transaction to request them.

The penalty you pay for THREADWAIT=YES is that you are delaying a transaction that might otherwise run immediately if allowed to create a new thread.

The penalty you pay for PROTECTNUM>0 is that thread resources (memory, plan locks, etc.) remain held even when no transaction is using it. This can cause contention with DB2 utilities which require exclusive access.

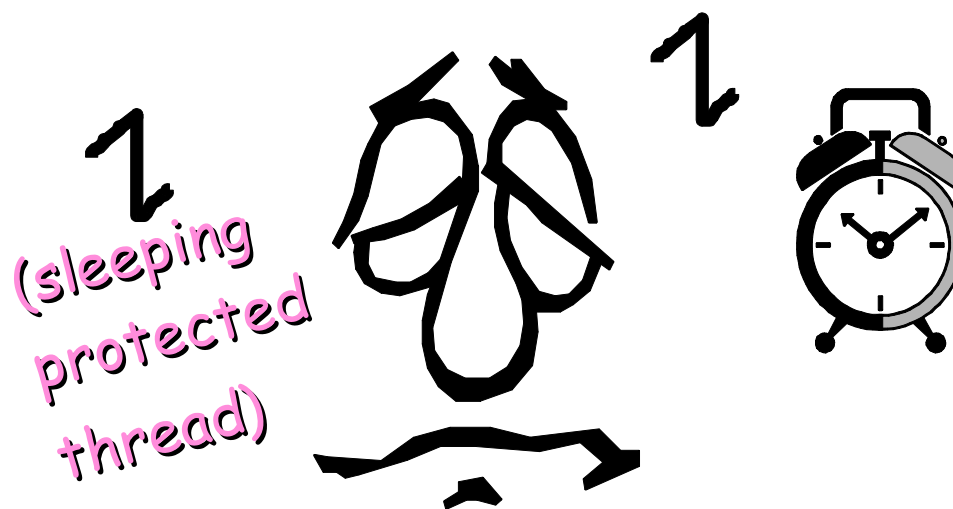
Protected Threads

Unprotected threads

terminate immediately upon release, unless another SQL request is waiting to use it
(THREADWAIT=YES)

Protected threads (PROTECTNUM)

once released, wait 2 consecutive purge cycles (default 30 seconds each cycle)
terminate if unused at the end of the purge cycle



Notes: Protected threads

Protected threads are indicated by **PROTECTNUM** on a DB2ENTRY. The pool does not have protected threads.

Protected threads will remain available for reuse after released for 2 purge cycles. The **PURGECYCLE** parameter on the DB2CONN sets this value. Thus, the average time an unused thread will remain allocated is $1.5 * PURGEC$.

Unprotected threads are indicated by `THREADLIMIT-PROTECTNUM`. So, for an entry, if `THREADLIMIT=5` and `PROTECTNUM=2`, there would be 2 protected threads and 3 unprotected.

Unprotected threads are terminated immediately upon thread release if no other task is waiting. This means that a task that releases its thread at an intermediate SYNCPOINT may have to go through thread creation multiple times within one transaction.

NOTE: Unprotected threads *can* be reused. It is a common fallacy that you must use a protected thread to get thread reuse. Thus, POOL threads can be reused though less likely if many different plans are executing in the pool.

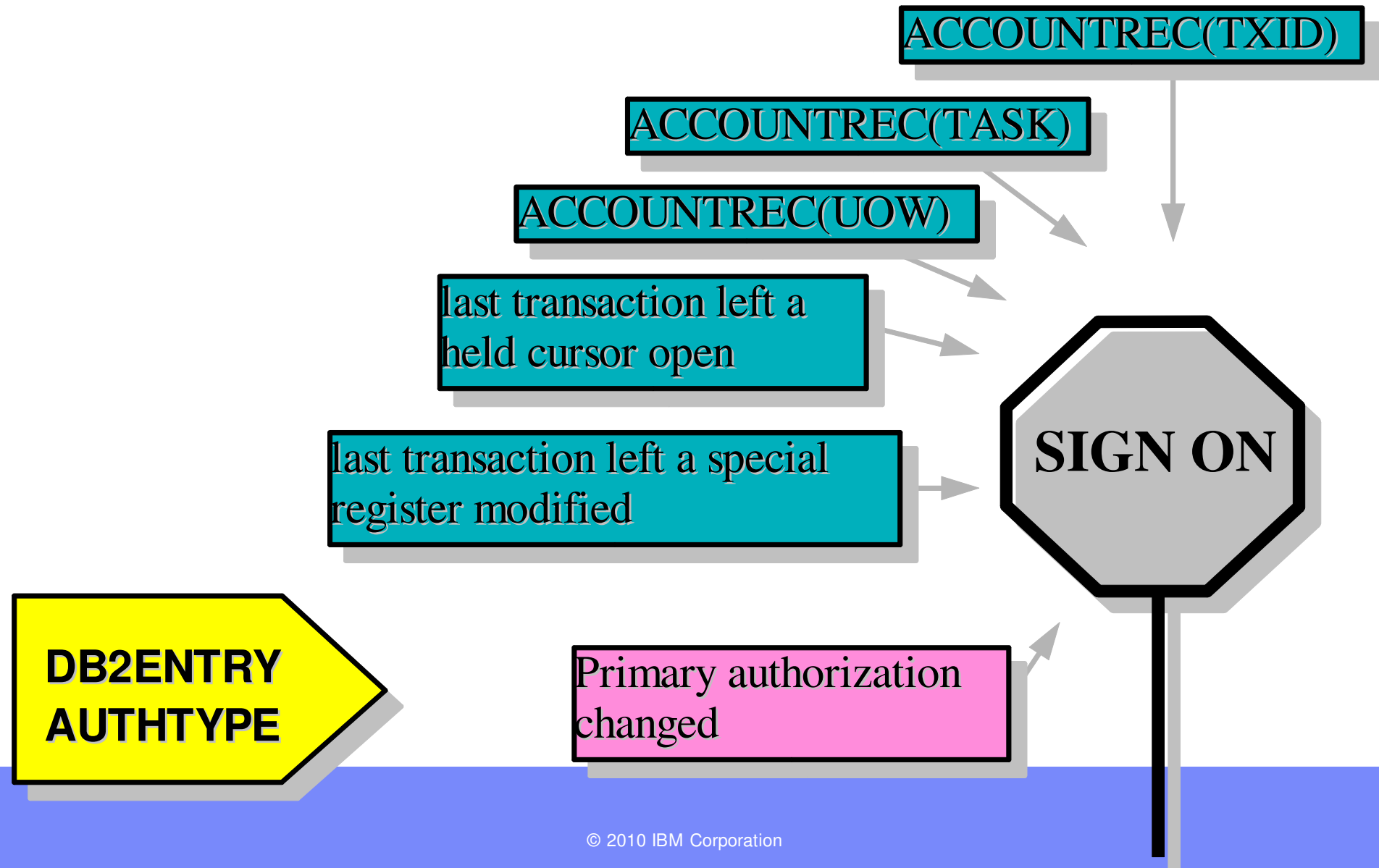
More on Reuse



- Avoid getting and releasing DB2 resources
 - ▶ Use ACQUIRE(ALLOCATE) and RELEASE (DEALLOCATE) parameters of BIND
 - ▶ Very efficient for DB2 data sharing
- Caveat:
 - ▶ May cause EDMPOOL to grow
 - ▶ Too much reuse can be a bad thing !
 - Very Long running threads become “fat”

- A transaction can save the cost of DB2 resource allocation by specifying ACQUIRE(ALLOCATE) and RELEASE (DEALLOCATE).
 - ▶ PRO: In a DB2 data sharing environment this provides significant reduction in messages to the coupling facility.
 - ▶ CON: Locking concurrency problems can occur if resources remain allocated for a long time.
 - ▶ CON: EDMPOOL will grow as more packages are used within the life of a thread.

Avoiding sign-on



Notes: Avoiding signons

Avoiding signons where ever possible will improve the performance of transactions within a DB2ENTRY

Sign on always occurs when a thread is used for the first time. A thread is signed on again when the thread is reused and any of the following occur:

- The primary authorization ID changes: The **AUTHID** or **AUTHTYPE** parameter value will greatly affect the frequency that the primary authorization changes

 - The best choice is a value that is constant such as a AUTHID(string) or AUTHTYPE(SIGN). The worst choice is a value that may change frequently such as AUTHTYPE(TERM) or AUTHTYPE(USERID).

- The first SQL statement after a SYNCPOINT if **ACCOUNTREC(UOW)** is specified. This provides the ability to correlate DB2 accounting with CICS accounting on a per UOW transaction basis. For transactions with multiple UOWs per transaction, multiple DB2 accounting records have to be correlated. With CICS TS 1.2 this can be overcome by specifying ACCOUNTREC(TASK).

- The TXID changes on a DB2ENTRY used by multiple transactions. This can be avoided by using ACCOUNTREC(TASK) or ACCOUNTREC(NONE)

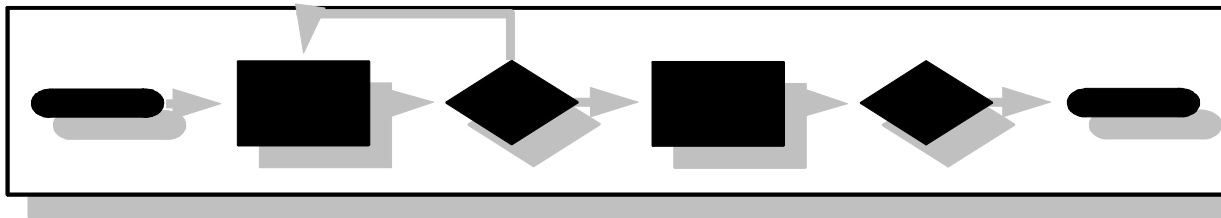
- The last transaction left an open held cursor or left a special register in a modified state.

If you do not use DB2 security and set grant access to PUBLIC, you should also specify CACHESIZE(0) for the BIND PLAN.

Signons done for accounting purposes only are called **partial signons**. Whereas DSNB DISPLAY STATISTICS only has an AUTH column which will include both full and partial signons, the enhanced CICS-DB2 statistics available via DFH0STAT or DFHSTUP will show the a Signon count (ie full and partial signons) and a partial signon count.

Good Application Design

- Code applications to make threads reusable
 - ▶ Close cursors when no longer needed
 - ▶ Restore special registers to their initial state
 - ▶ Issue SYNCPOINTS as soon as practical
- Use fewer plans
 - ▶ Bind packages into single plan to allow DB2 to manage the packages in memory and increase thread reuse

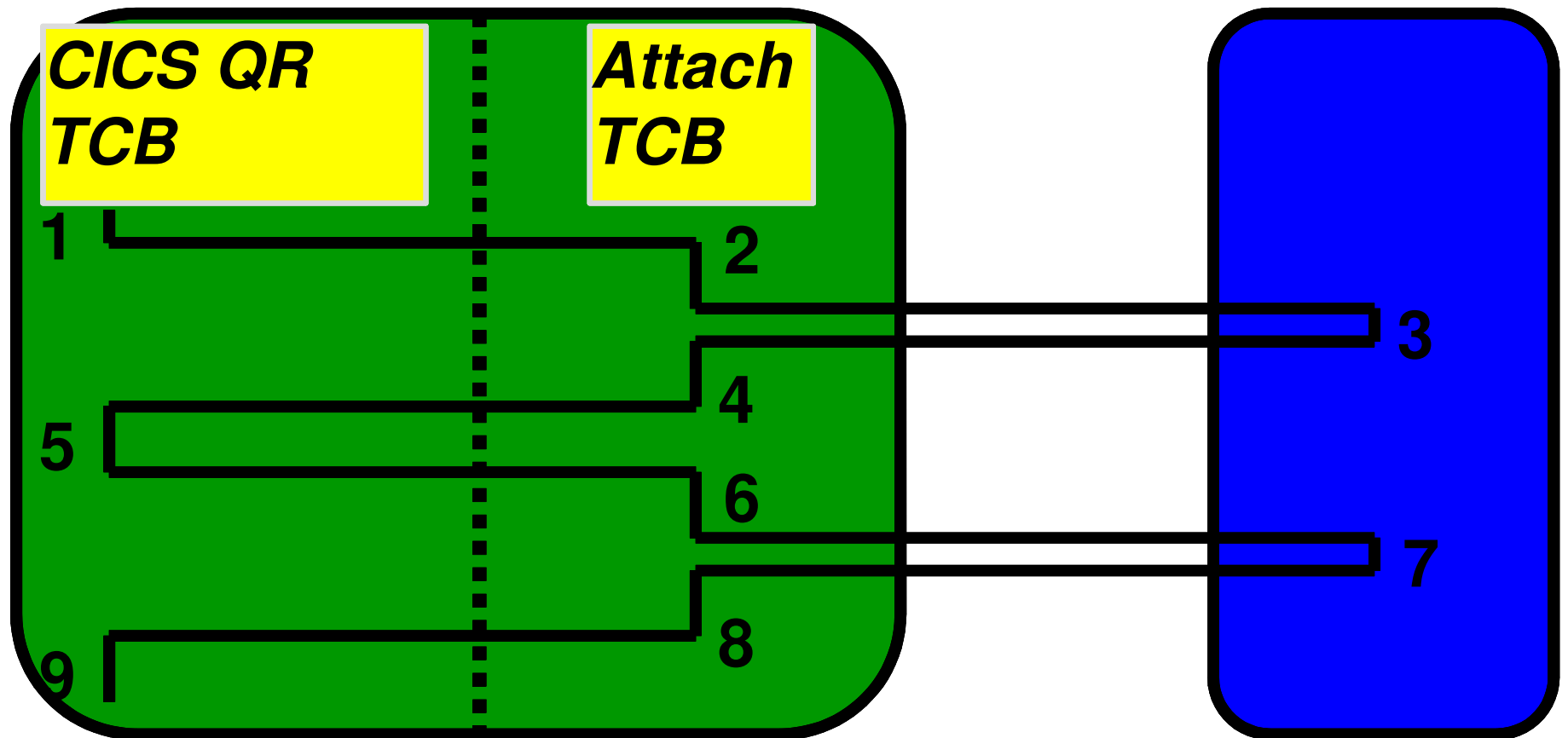


- When a thread has open held cursors or modified special registers, the thread cannot be reused at intermediate SYNCPOINTS (i.e. before EOT).
 - ▶ It is a good practice to close all cursors, especially those using the WITH HOLD option, when they are no longer needed to release locks and make the thread reusable. The extra effort it takes to restore special registers to the initial value may well be worth the performance increase achieved by making the thread reusable.
 - ▶ If the application does not close cursors or restore special registers, a partial SIGN-ON will occur to restore the thread to the initial state at EOT.

- Use a single large plan with many packages rather than using Dynamic Plan Switching PLANEXITNAME(nnn) (RCT PLNEXIT=YES) to switch between many smaller plans.
 - ▶ Dynamic Plan Switching (DPS) reduces the possibility of reusing an existing thread. Also, a transaction can only switch plans when the thread is reusable.
 - ▶ Packages definitions should be tuned. List most frequently used packages first and use wildcarding where possible.
 - ▶ A large collection does not present a significant performance impact.

A reminder about accounting prior to OTE...

CICS Address Space



CICS CMF CPU = 1+5+9

DB2 Class-1 CPU = 2+3+4+6+7+8

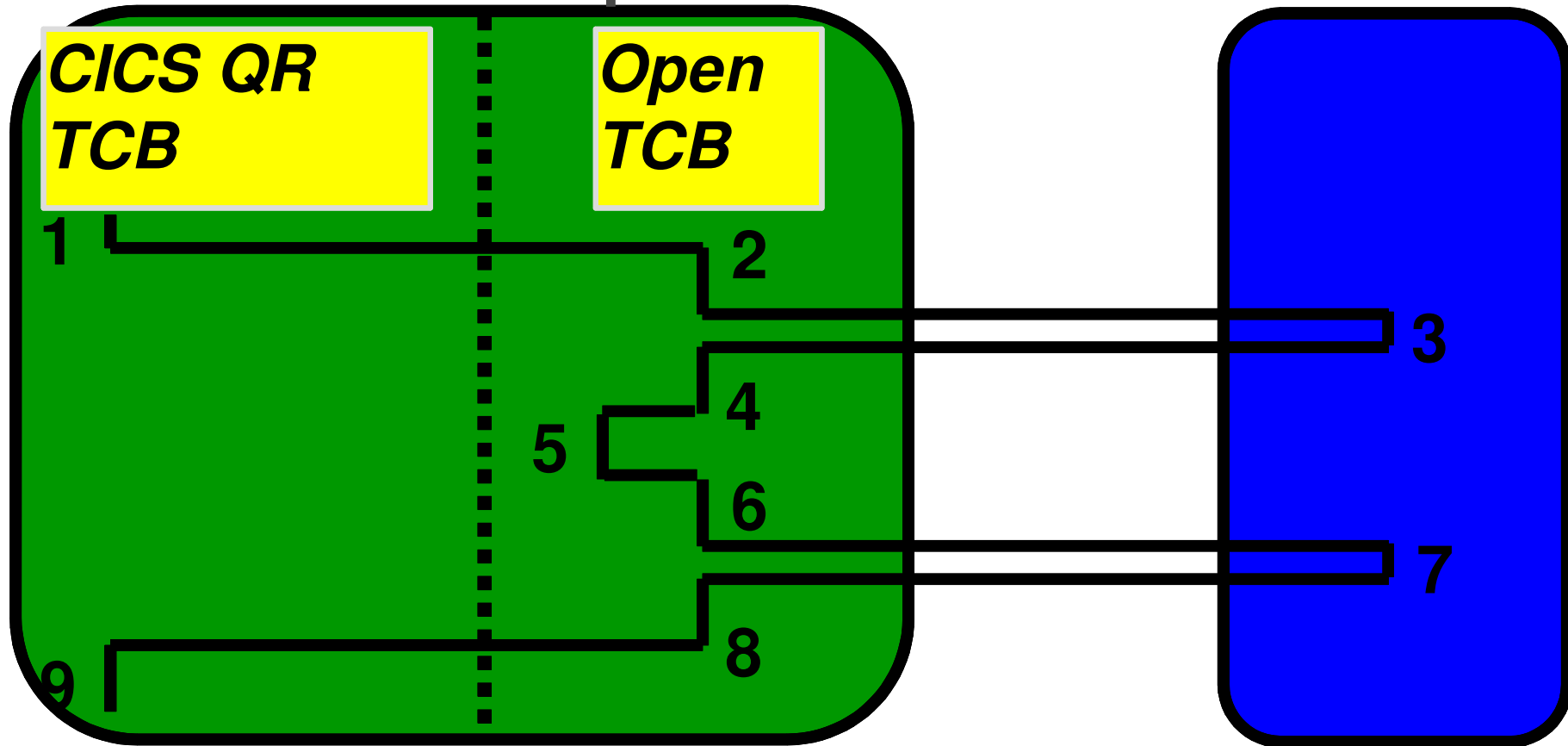
Class-2 CPU = 3+7

Notes

- The picture shows how processor time was recorded for a CICS-DB2 application running in CICS TS 1.3 prior to use of OTE. The same applies to CICS TS 2.2 when connected to DB2 V5 or below, ie when OTE is not being used.
- The CICS Monitoring Facility (CMF) will record the cpu time consumed on the CICS QR TCB for transaction attach, detach, time spent processing CICS commands and time spent in the application.
- DB2 Class 1 cpu time records the time spent on the CICS-DB2 Attach tcb which includes both time spent in the CICS address space in CICS-DB2 Attach code and time spent in DB2 address spaces.
- DB2 Class 2 cpu time is a subset of DB2 Class 1 cpu time, and just records the cpu time spent in DB2 address spaces.

Accounting in an OTE environment

CICS Address Space



CICS CMF CPU = 1+2+3+4+5+6+7+8+9

DB2 Class-1 CPU = 2+3+4+5+6+7+8

Class-2 CPU = 3+7

Notes

- The picture shows how accounting changed when the CICS-DB2 Attach exploited OTE.
- The CICS Monitoring Facility (CMF) now reports the cpu time for the whole application, that is attach, detach, application code and the time spent both in the CICS-DB2 Attach and time in the DB2 address spaces.
- CMF cpu time is reported both as an overall figure across all CICS TCBs (including open tcbs) and also broken down into different TCBs (QR, L8 open tcb, RO TCB - if the program had to be loaded first for example)
- Cpu time 5 represents time spent in the application in between DB2 calls. If the application is threadsafe then cpu time 5 will be consumed on the open tcb as shown in the diagram. Note also that it will also be recorded as part of DB2 class 1 time.
- If the application is not threadsafe, then cpu time 5 will be consumed on the CICS QR TCB. and will not appear as part of DB2 class 1 time.
- The DB2 class 1 cpu time (which includes DB2 class 2 cpu time) is included in the cpu time recorded by CICS. **When using DB2 V6 or higher you no longer need to add DB2 class 1 cpu time to the CICS reported cpu time (irrespective of whether the application is threadsafe or not) else you will double account the DB2 cpu time.**

Accounting in an OTE environment

- DB2 Class 1 time will be included in the CICS cpu time
- L8 cpu time can be greater than DB2 class 1 time
 - ▶ May also contain thread create/termination time.
 - ▶ If application is threadsafe ...
 - will contain cpu time spent in application
 - QR cpu time will decrease
- DB2WAIT field will be zero
 - ▶ represents elapsed time spent waiting for a DB2 request to complete.
 - ▶ with OTE there is no CICS dispatcher wait for a subtask.
- Can be large difference between DB2 Class 1 and Class 2 cpu times
 - ▶ CICS RMI code and threadsafe application code
 - ▶ CICS tracing

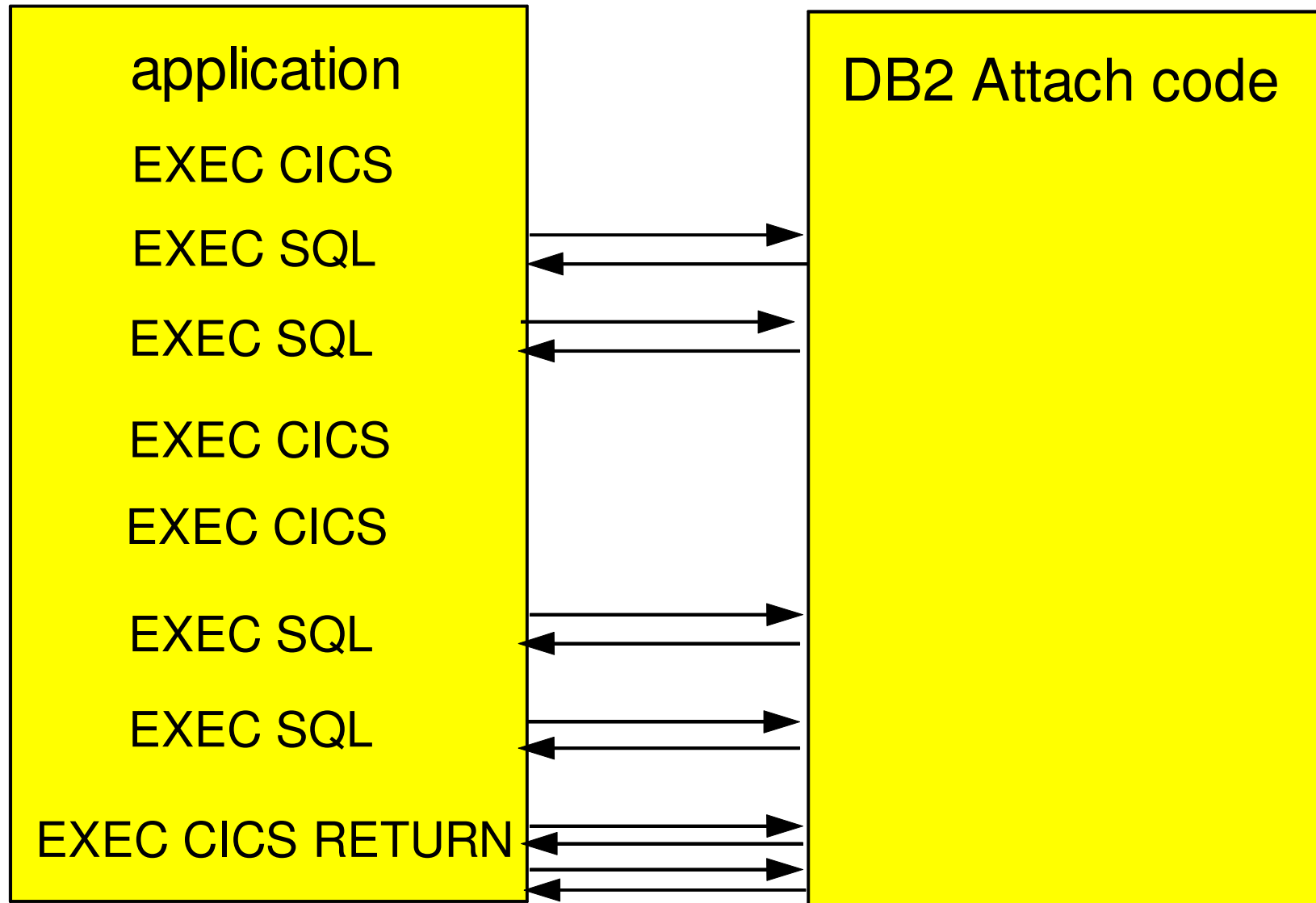
Notes

- In this new CICS TS 2.2 OTE environment, the total cpu time reported by CICS for an application may be higher than that reported previously in CICS TS 1.3 as result of adding the CICS and DB2 class 1 cpu times together. This is because:
 - If execution of the application caused the CICS-DB2 Attach to create a thread, then in CICS TS 1.3 most of the thread creation cost was not reported by DB2 Class 1 cpu time, nor by CICS. Now it will be recorded as part of the L8 open tcb cpu time.
 - Similarly thread termination costs will now be recorded as part of the L8 cpu time.
- Do not be surprised to see the CICS monitoring DB2WAIT value reported as zero. This represents the elapsed time the CICS-DB2 Attach was in a CICS dispatcher wait, whilst waiting for the request to be serviced in DB2. This now only applies when using DB2 V5 and below. With OTE, there is no cics dispatcher wait in the CICS-DB2 Attach. The Attach is called on an L8 open TCB and it calls DB2 running on the same L8 open TCB. If you want to see the wait time inside DB2, you will have to look at the Class3 suspend time in the DB2 accounting records. CICS Performance Analyzer (see Session 1040) gives you an easy way of correlating CICS and DB2 accounting records.
- Previously, the difference between db2 class 1 cpu time and DB2 class 2 cpu time was the cpu time spent in the CICS-DB2 Attach, and this was quite small. Do not be surprised if there is a large difference now. Not only will DB2 class 1 time now include time spent in the application (if the application is threadsafe), but more significantly DB2 class 1 time can include CICS tracing. Previously, we could not use CICS trace facilities in the CICS-DB2 subtask. Now if RI (RMI) level 2 trace is active we will trace before and after all calls to DB2. This will be included in the DB2 Class 1 cpu time.

CICS-DB2 TCB switching prior to CICS TS 2.2

QR TCB

DB2 Attach TCB



Notes: CICS-DB2 TCB switching prior to CICS TS 2.2

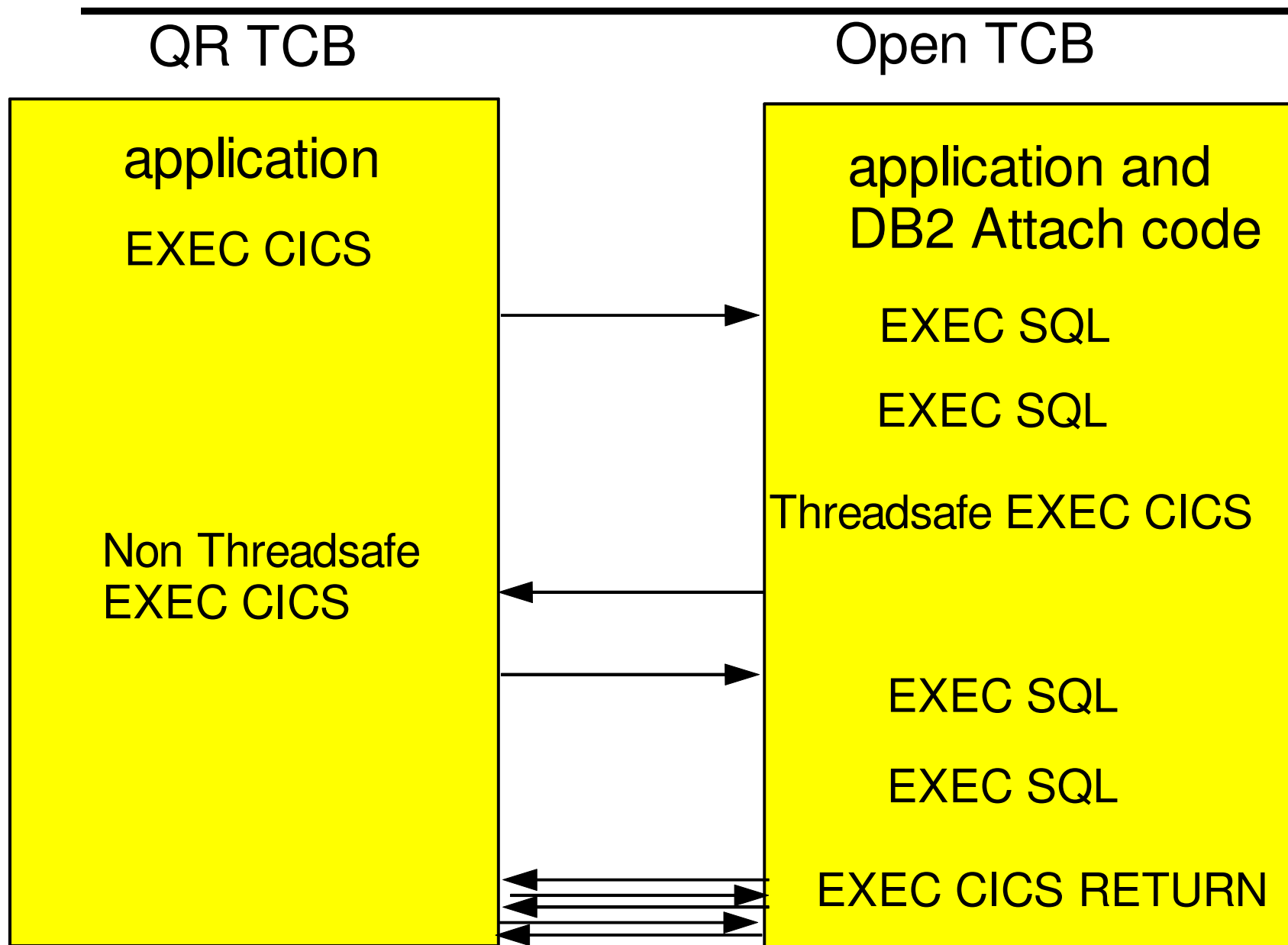
The diagram shows what happens today when running a CICS DB2 application containing both EXEC CICS commands and EXEC SQL commands. In particular it is showing the TCB switching that goes on when executing DB2 calls.

The CICS-DB2 Attach has to manage its own set of private TCBs onto which is offloaded the DB2 request. The DB2 Attach TCB is the one under which calls are made to DB2 as we cannot use the QR TCB, as for example I/O requests would halt the QR TCB (and so the whole of CICS) whilst the I/O was satisfied. This is called blocking the TCB.

For each EXEC SQL request there are two TCB switches, one to offload the request onto the DB2 Attach TCB, another TCB switch to return control to the application on the QR TCB when the DB2 request has completed.

In total there are twelve TCB switches (four for the two phase commit end of task syncpoint). Contrast this with the next page showing how the CICS-DB2 Attachment facility exploits OTE in CICS TS 2.2 with DB2 V6 or higher.

TCB switching for a threadsafe application in CICS TS V3 & V4



Notes: TCB switching for a threadsafe application in CICS TS V3 and V4

The picture shows the result of upgrading the CICS-DB2 Attachment facility in CICS TS 2.2 to take advantage of OTE and use an open TCB to call DB2 (when using DB2 V6 or higher) rather than one of its own privately managed TCBs. As a result of connecting to a DB2 V6 or higher system, the CICS DB2 TRUE is enabled with the OPENAPI keyword. Let us assume that the calling application is threadsafe and been defined as such on the program definition.

The first EXEC CICS command is executed under QR. 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. For the next EXEC SQL request we stay on the open TCB. Next is a threadsafe EXEC CICS command, so we can stay on the open TCB. Next is a non threadsafe command and so we have to switch to QR to execute that request. For the next SQL request we return to the open TCB. For EXEC CICS RETURN we return to QR TCB followed by the four TCB switches for the syncpoint two phase commit protocol.

It can be seen that the number of TCB switches is reduced in this environment. Avoid interspersing CICS commands that are not threadsafe between EXEC SQL commands as this will cause excessive TCB switching.

Whilst in this simple example we only reduce the number of TCB switches from 12 to 8 the reduction is more dramatic when you consider a DB2 application that does 100 EXEC SQL requests that today would incur 204 TCBs switches. This can be reduced to 6 TCB switches!

Support for Group Attach

DB2GROUPLD parameter added to DB2CONN and INQUIRE/SET DB2CONN commands in CICS TS 2.2

Mutually exclusive to DB2ID parameter

CICS will connect to any member of the group

DB2 searches list of subsystems defined to this MVS

RESYNCMEMBER(YES|NO) parameter added to DB2CONN

YES (default) means if CICS thinks indoubts are outstanding then ignore group attach and attach to the specific DB2 member last used

NO means use group attach regardless of indoubts

CICS will nevertheless try a specific attach to the last used group member first. If that fails it will use group attach to connect to any group member.

Notes: Support for Group attach

Group Attach is activated by specifying a data sharing group id in the DB2GROUPLD parameter in the DB2CONN definition. DB2GROUPLD is mutually exclusive with the DB2ID parameter.

When using group attach, DB2 will connect CICS to any one of the data sharing members active on the MVS image. Note it still has to be active on the same MVS image where CICS is running, group attach is not affecting that basic premise.

If more than one member of the data sharing group is active on the MVS image, then DB2 will pick a subsystem to connect to.

Pick unique group names. DB2 allows group names to be the same as a member name. I would not recommend this.

DB2 does not have peer recovery. That is, one DB2 cannot initiate recovery of indoubts created when CICS was connected to a different DB2. Hence using group attach can exacerbate this problem. Indoubts created when attached to one member will not be resolved until reconnection to the same member.

CICS provides the ability for the user to specify that CICS should override group attach, if it believes it is holding UOW resolutions for indoubt units of work from the last connected DB2. CICS cannot be 100% sure DB2 is indoubt about the UOWs it is holding (eg DB2 might have got the commit, but the system went down before DB2 could tell CICS to forget the UOW). CICS will assume DB2 is indoubt about the UOWs.

Support for DB2 V8 enhanced Restart-Light

- Restart-Light was introduced in DB2 V7
 - ▶ Shutdown DB2 with reduced storage footprint
 - ▶ DB2 restarts to flush out in-flight UOWs (to minimise retained locks) then terminates
- DB2 V8, V9, V10 Restart light - subsystem remains active if indoubts outstanding
 - ▶ Flushes out in-flight UOWs
 - ▶ Awaits resynchronisation of indoubts
 - ▶ With later releases is now configurable
- CICS connects to DB2, DB2 informs CICS it is restart-light subsystem
 - ▶ CICS-DB2 Attach connection status remains as 'connecting'
 - stops any new work accessing DB2
 - ▶ CICS allows resynchronisation tasks (CRSY) to access DB2
- DB2 Restart-light system terminates when resync complete

- DB2 restart-light introduced in DB2 V7 is a cutdown DB2 subsystem with a light storage footprint. Its is useful in a cross system restart in the event of a MVS system failure. Its flushes out retained locks where possible to allow for faster recovery and data availability. However in DB2 V7 it does not deal with retained locks due in indoubt units of work.
- DB2 V8 enhanced restart light so that having released retained locks for in-flight units of work, instead of immediately terminating, the restart light system will remain active if indoubt units of work are present. This allows CICS to reconnect and resynchronise those units of work, before the restart light system terminates. It is useful in overcoming the fact that DB2 does not have peer recovery meaning one DB2 subsystem cannot recover the indoubts on behalf of another DB2 subsystem. Consider the following scenario :

DB2A and DB2B are members of datasharing group FRED

CICSA is connected to DB2A via group attach to group FRED on LPAR 1

CICSB is connected to DB2B via group attach to group FRED on LPAR 2

LPAR 1 fails causing indoubt UOWs in DB2A

CICSA is restarted on LPAR2. DB2A is restarted in light mode on LPAR2

CICSA uses group attach but with RESYNCMEMBER(YES) meaning that because indoubt UOWs are outstanding, group attach is ignored and CICSA insists on reconnecting to DB2A. DB2A initiates restart-light. Resynchronisation takes place, then DB2A terminates.

CICSA drops into standbymode and tries to reconnect. Now that no indoubts are present, group attach is used and CICSA connects to DB2B and normal processing can now continue.

Tuning summary

- Set Thread and TCB Limits
 - CTHREAD = FACTORED TSO/BATCH + TCBLIMIT(DB2CONN)
 - SIT: MAXOPENTCBS. Note: the default is 12 which will be too low for a DB2 workload
- KEEP IT SIMPLE
 - Use the POOL until a need for DB2ENTRY is established
- DB2ENTRY
 - Isolate High PRIORITY OR High Volume transaction
 - Special Transaction Requirements
 - Authorization
 - Plan Selection
 - PRIORITY
- Review CICS-DB2 statistics

Notes: Tuning Guidelines

MAXOPENTCBS in SIT: When CICS is connected to DB2 V6.1 or above, CICS creates L8 TCBs up to a limit of MAXOPENTCBS in the SIT. The TCBLIMIT (DB2CONN) specifies how many of these may be used to access DB2.

Session References:

- DB2 Administration Guide - SC26-9931
- DB2 Command Reference - SC26-9934
- DB2 Packages: Implementation and Use - GG24-4001
- DB2 Application Programming Guide - SC26-9933
- CICS TS CICS DB2 Guide - SC34-5707
- CICS TS Resource Definition Guide - SC34-5990

CICS Explorer

- With CICS TS 4.1 there is full support for CICS-DB2 definitions via CICS Explorer
 - Can work at a CICSPLEX level utilising CPSM
 - Can run it against a single CICS Region
 - CICS Explorer is the new face of CICS
- CICS IA plugin to Explorer helps with identifying non threadsafe applications
- CICS PA plugin with Explorer helps with performance analysis and benchmarking

Queries Regions

- Specific
 - Threadsafe
 - All programs that issue a GETMAIN
 - All programs that issue an ADDRESS
 - All programs that issue an EXTRACT
 - All programs that issue an LOAD
 - All programs which may have thread
 - CICS commands by TCB mode and pr
 - DB2 commands by TCB mode and pr
 - IMS commands by TCB mode and pr
 - MQ commands by TCB mode and pro
 - Webservices
- DB2
- IMS

*Resources

All programs which may have threadsafe data integrity issues (29)

- PROGRAM (CCVSMGSF) (1)
- PROGRAM (OISA4000) (1)
- PROGRAM (CCVSMCSD) (1)
- PROGRAM (CCVSMMSGH) (2)
 - LOAD (1)
 - Resource Type (PROGRAM) (2)
 - PROGRAM (CCVSLITT)
 - PROGRAM (CCVSMSGT)
 - GETMAIN (1)
 - Resource Type (STORAGE) (1)
 - STORAGE (ADDR)
- PROGRAM (CCVSOWAB) (1)
- PROGRAM (CCVWSSH) (1)
- PROGRAM (CCVWSDSH) (1)

Uses

Program(CCACMDA) in Region CICACB25 (50)

Resources used	By Resource
Program (23)	
UOW (1)	
(1)	
STORAGE (1)	
File (9)	
TD (1)	
TSAUX (4)	
TS (4)	
ENQNAME (5)	
STORSHR (1)	

Programs Transactions

* in Region CICACB25 (38)

- CCVACMDA
- CCVACRE
- CCVADISP
- CCVAETIM
- CCVAINQ
- CCVALIST
- CCVAUPD
- CCVSARC
- CCVSCXTC
- CCVSEXP
- CCVSIMP
- CCVSLITT
- CCVSMCSD
- CCVSMDDD

Used By

Programs using EXIT(CCVIANCH) in All regions (4)

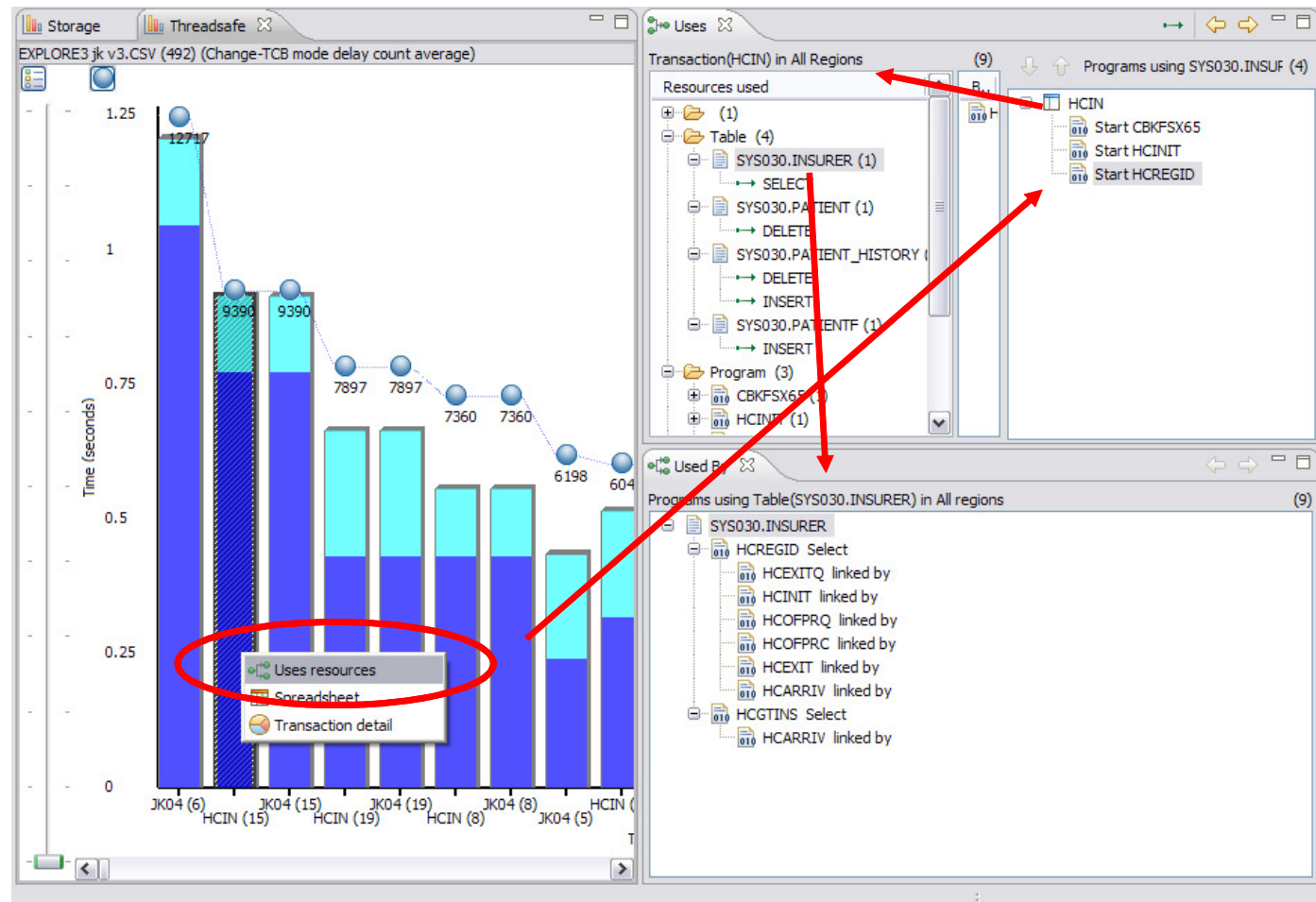
- CCVIANCH
 - CCVSWAKE Extract
 - CCVWSDSH Extract
 - CCVSWASH Extract
 - CCVADISP Extract

(150)

- CCVACMDA
 - Link CCVACRE
 - Link CCVSEXP
 - Link CCVSMMSGH
 - Link CCVSUTIL
 - Link CCVAINQ
 - Link CCVALIST
 - Link CCVAUPD
 - Link CCVSMMSGH

Threadsafe (OTE) – Application analysis

- Use a combination CICS Explorer, CICS IA and CICS PA to see the whole picture



Further Information

- CICS TS 3.1/3.2/4.1 Information Centers
 - Refreshed regularly, can be downloaded from:
 - <http://www.elink.ibm.link.ibm.com/public/applications/publications/cgibin/pbi.cgi>

- **Redbook: Threadsafe considerations for CICS SG24-6351-00**
 - <http://www.ibm.com/redbooks>
 - Third edition (November 2007) incorporates CICS TS 3.2 enhancements

- CICS Tools to help threadsafe migration
 - <http://www.ibm.com/cics/tools>