CICS/RLS STUDY

Authors

John Burgess, IBM
Arndt Eade, IBM

First edition (January 2010)

This edition applies to Version 1.0 of "SupportPac CP13 – CICS RLS Study" and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Table of contents

Figures

Notices

The provisions set out in the following two paragraphs do not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

Information contained and techniques described in this publication have not been submitted to any formal IBM test and are distributed on an "AS IS" basis.

The use or implementation of any information contained and/or of any technique described in this document is the user's responsibility and depends on the user's ability to evaluate and integrate the information and/or technique into the user's operational environment. While IBM has reviewed each item for accuracy in a specific situation, IBM offers no guarantee or warranty that the same or similar results will be obtained elsewhere. Users attempting to adapt any technique described in this document to their own environments do so at their own risk.

The information contained in this publication could include technical inaccuracies or typographical errors.

Changes are periodically made to the information contained herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any reference in this publication to an IBM licensed program or another IBM product is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe applicable intellectual property rights may be used instead of the referenced IBM licensed program or other IBM product.

The user is responsible for evaluating and verifying the operation of the material supplied in conjunction with this publication in conjunction with other products, except those expressly designated by IBM.

International Business Machines Corporation may have patents or pending patent applications covering subject-matter described in this document. The furnishing of this document does not give you any license to any such patent. You can send license inquiries, in writing, to:

The IBM Director of Licensing
International Business Machines Corporation
North Castle Drive

Armonk, NY 10504-1785
U.S.A.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

z/OS®, zSeries®, z10

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of
Microsoft Corporation in the United States, other countries, or both.

UNIX is a trademark of The Open Group in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Disclaimer

Information contained in this report has not been submitted to any formal IBM test and is distributed “as is”. The use of this information and the implementation of any of the techniques is the responsibility of the customer. Much depends on the ability of the customer to evaluate this data and project the results to their operational environment.

The performance data contained in this report was measured in a controlled environment and results obtained in other environments may vary significantly.

Updates

Date (MM-DD-YY)	Version	Change
01-03-2012	V1	- Correct SOC7 abend in SMF42PRC due to incorrect data type definition - Correct missing version number in SMF product level header reporting. Notes: Object deck compile date of 20111220-145144

SupportPac files

This SupportPac contains the following artifacts:

License Directory - Directory of license information.
SupportPac CP13.pdf - This document
SupportPac CP13.zip - Zip file containing two partitioned datasets in sequential form

Installing the SupportPac

Included in the SupportPac are two partitioned data sets that contain sample JCL and programs, in object code format, to format and print SMF 42 subtype 16-19 records.

There are two files called CP13.*jcl* and CP13.*obj* contained in CP13.*zip*. These files need to be transferred to the destination TSO system as sequential binary file with a record format of FB 80.

Send CP13.JCL to TSO with a name CP13.JCLSEQ.
Send CP13.OBJ to TSO with a name CP13.OBJSEQ.

Use one of the following methods to accomplish this:

- To send them via ftp ensure the BINARY option is set then use the following commands

  quote site fixrecfm 80 (optional)
  Put CP13.JCL CP13.JCLSEQ

- With Personal Communications, use the *Send Files to Host* option under the Transfer menu item to transmit to TSO

  PC File CP13.JCL
  Host File CP13.JCLSEQ
  Transfer Type PDS

The Transfer type of *pds* may need to be correctly Setup. To do this, use the *Setup, Define Transfer Types* option under the Transfer menu item and create the *pds* type with the *ASCII, CRLF* and *Append* checkboxes all unselected, the *Fixed* radio button selected and the *LRECL* set to 80.

Repeat with the CP13.OBJ file.

On TSO, issue the command: *receive indsname(CP13.JCLSEQ)* to unload the sequential files into a TSO partitioned dataset. When prompted for a filename, reply: *dsn(CP13.JCL).* Repeat the step for CP13.OBJSEQ.

CP13.JCL contains the following members:

Member	Description
LINKEDIT	Sample JCL to link edit the SMF42PRC and TIMECONV object members into an executable module called SMF42PRC. The JCL requires customisation prior to submission. See instructions within the comments of the JCL for details.
SMF42PRC	Sample JCL to execute the SMF42PRC sample program. This program will read SMF data and format the contents of any SMF type 42 subtype 16-19 records found.

CP13.OBJ contains the following members:

Member	Description
SMF42PRC	Object code of program to produce formatted reports of SMF 42 subtype 16-19 records
TIMECONV	Formatting program used by SMF42PRC

Once the PDS datasets have been created you should perform the following steps:

1) Review the IBM licensing agreement which is included in the zip directory.
2) Customise and execute member CP13.JCL(LINKEDIT) to create an executable load module
3) Customise and execute member CP13.JCL(SMF42PRC) to create the formatted reports

Introduction

The objective of this study is threefold:

1. Show comparisons and performance characteristics of a simple workload using:
   - Local LSR VSAM files
   - Cross Memory MRO Function Shipping (FS)
   - XCF MRO FS
   - RLS
2. To outline and explain the CICS and RLS activities that occur when a CICS program issues the various API calls to RLS - these being: READ, READ UPDATE, REWRITE, ADD, BROWSE and DELETE. This is mainly from a performance perspective but also to give the reader a better understanding of the processing that occurs when a RLS call is made.
3. To document how monitoring data - generated by CICS, RMF and SMS - can be extracted and correlated along with some key fields that can be examined to indicate whether configuration or usage patterns can be optimised.

To achieve the above we start by giving an overview of RLS and its components - in CICS & RLS - Concepts, terms and components – and then follow with some CICS migration scenarios that explain why customers should consider migrating to SMS managed RLS datasets.

To assist in the understanding of the SMF 42 records produced by RLS it was necessary to write a formatting program. Sample output from this program has been included in this report. The program, in object code format is being made available to customers as part of this SupportPac. Instructions on link-editing and running the program can be found by following this link SMF 42 formatter reports

CICS & RLS - Concepts, terms and components

What is RLS?

RLS is a VSAM function first provided by DFSMS/MVS 1.3 and exploited by CICS TS 1.1 and above.

VSAM data sets are opened in RLS mode, which allows them to be shared, with full update capability, among applications running in multiple CICS regions within a Sysplex.

Back in DFSMS/MVS 1.3, support started for a new data sharing subsystem, SMSVSAM, which runs in its own address space. SMSVSAM provides the VSAM RLS support required by CICS application-owning regions (AORs) and batch jobs within each MVS system image in a Parallel Sysplex environment.

The SMSVSAM subsystem, which is generally initialized automatically during an MVS initial program load (IPL), uses the coupling facility for its cache and lock structures. It also supports a common set of buffer pools for each MVS image.

CICS and RLS

Prior to RLS, CICS users had been able to share VSAM data sets with integrity by using function shipping to a File Owning Region (FOR). With function shipping, one CICS region accesses the VSAM data set on behalf of other CICS regions. Requests to access the data set are shipped from the region where the transaction is running to the region that has access to the file. Function shipping provides a solution for the CICS user, but it does have limitations: Function shipping does not address the problems of sharing data sets between CICS regions and batch jobs. It should also be noted that the FOR is constrained to a single TCB and therefore the speed of a single CP. RLS is not subject to this constraint as all of the work is done across multiple Application Owning Regions (AORs) where the application resides. These RLS requests like LSR are capable of running in Threadsafe mode in which case they run on L8/9 TCBs. In non-Threadsafe mode the RLS requests run on SRBs within the CICS address spaces.

Buffer Coherency - RLS uses the buffer registration and invalidation functions of the coupling facility cache as the means to maintain buffer coherency across the local buffer pools. RLS uses the conditional write function of the coupling facility cache to implement an optimistic serialization protocol for changing data control intervals. The overall process is referred to as "record merge redo."
Locking - RLS locking is at the granularity of individual records and this permits multiple concurrently executing transactions to change different records that reside within the same data control interval (CI). RLS assigns a separate buffer containing a copy of the data CI to each of the sharers. The coupling facility local cache invalidate and conditional write functions are used to detect concurrent write activity at the CI level. If the first write is successful, it invalidates the buffers assigned to the other sharers. This causes their subsequent attempts to write to their buffers to fail. When RLS detects this failure, it internally re-accesses the dataset and reapplies the change. This process merges the change with the earlier changes and is the record merge redo.
DASD write serialization – RLS uses the Castout lock functions of the coupling facility to achieve this serialization.
Control interval/area split serialization - RLS uses an exclusive lock to serialize CI and CA processing for a KSDS. RLS uses the CIDFBUSY flag private serialization mechanism of the VSAM KSDS architecture to serialize requests to access data CIs while they are being split or moved by a control area split.

The following figure shows a typical configuration where CICS regions, running in multiple LPARs of a Sysplex are accessing files using Function Shipping (FS). The CICS region that owns and accesses the file(s) is commonly known as the File Owning Region (FOR) while the CICS region that hosts the application logic, and invokes function shipping requests to access the files, is known as Application Owning Region (AORs).

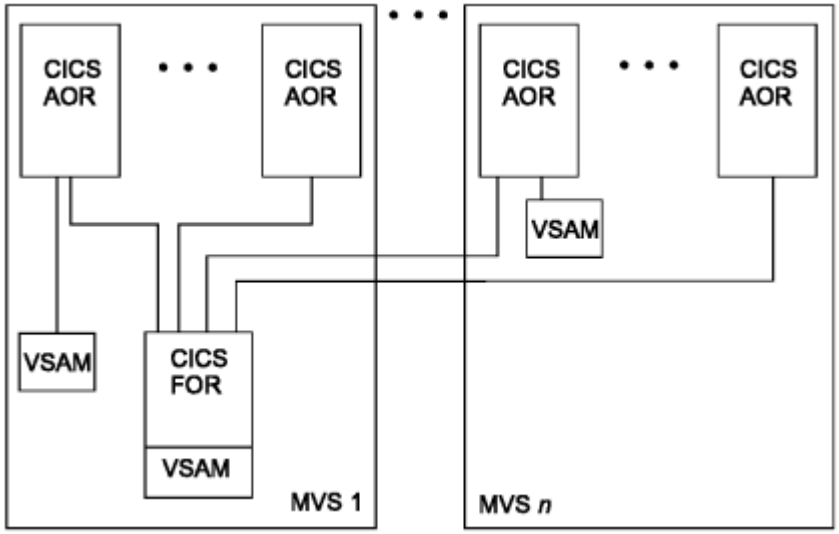


Figure 1.Sample CICS FS configuration

The following figure shows the same configuration where the AORs have been defined to access the files directly using VSAM RLS. Notice that the FOR is no longer required.

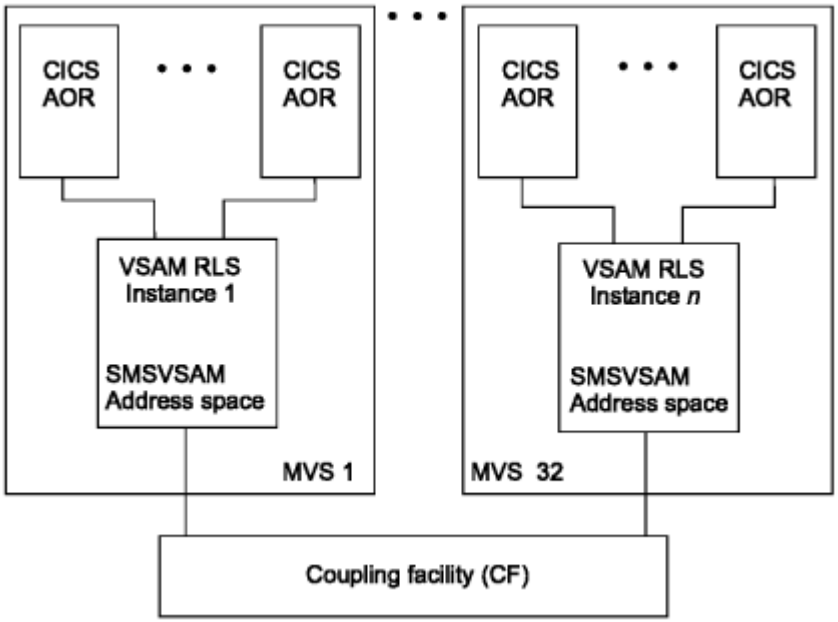


Figure 2.Sample CICS RLS configuration

Components of RLS

In order to coordinate and control access to VSAM files at a record sharing level, DFSMS, through the SMSVSAM address space, uses a number of components and facilities.

Each LPAR where RLS access is required will have an SMSVSAM address space active. This address space manages all access to datasets that are opened in RLS mode. To achieve this, SMSVSAM communicates through the use of XCF groups and CF cache and lock structures with SMSVSAM address spaces running on other LPARs within the Sysplex.

In order for a VSAM dataset to be opened for RLS access it must be SMS managed. Part of the process of defining the dataset as SMS managed requires, among other things, a Storage Class to be assigned. The Storage Class definition will contain a cacheset name with a number of CF Cache structures being assigned to the named cacheset.

In addition to the cache structures used by SMSVSAM as intermediate storage between local memory and DASD, a lock structure, called IGWLOCK00, is used to provide Sysplex-wide locking at the record-level. (In z/OS v1R10 up to 10 additional lock structures, called secondary lock structures can now be defined. These lock structures, which only contain record locks, will provide better separation of workloads, better CF balancing and availability.

Correct sizing of the CF cache and lock structures used by SMSVSAM is key to achieving optimum performance. You are strongly urged to consult the ***z/OS DFSMStvs Planning and operating guide SC26-7348*** for recommendations on the correct sizing of these structures.

The figure below shows a Sysplex configuration with a typical CICS and SMSVSAM configuration.

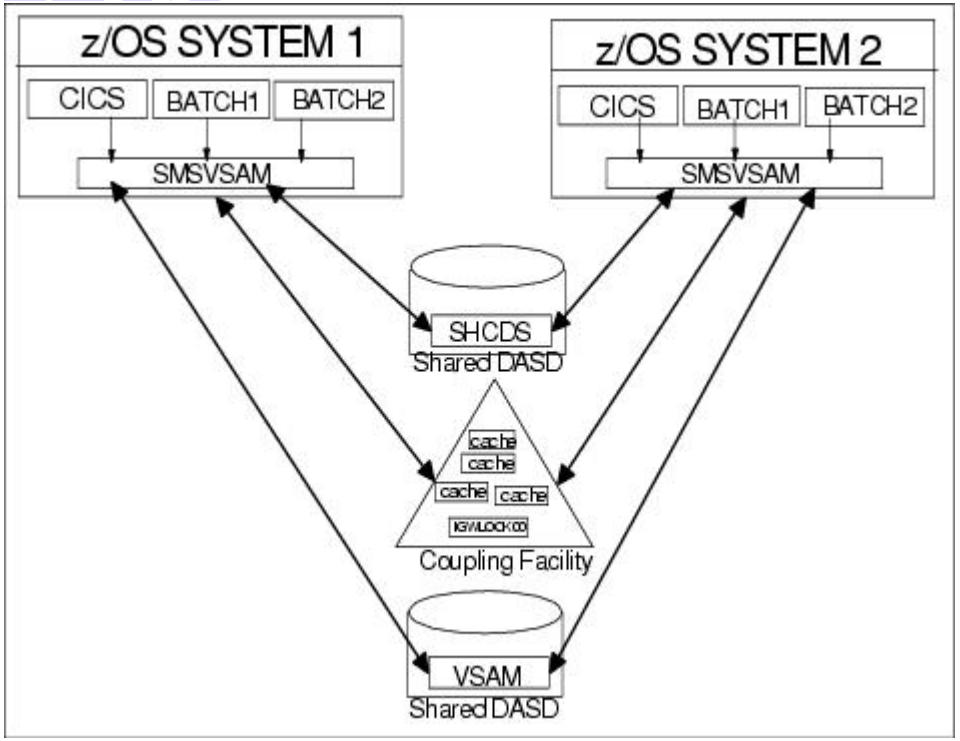


Figure 3.CICS and SMSVSAM configuration

What happens when a file access occurs?

This section outlines the processing that occurs for each type of RLS file access that can be made from a CICS application program through the CICS API. For each API call the possible paths are outlined along with any factors that can significantly affect the processing that occurs.

EXEC CICS READ and BROWSE (READNEXT and READPREV)

The processing that takes place when a single read, whether that be as a result of an EXEC CICS READ or an EXEC CICS READNEXT as part of a browse operation, is essentially the same.

A cross memory call is made from CICS to SMSVSAM requesting the record.

SMSVSAM first checks its local buffer pools for the record. If the record is currently within the buffer pool and still valid (*), then the data is returned and the request satisfied. If the record is not found in the local buffer pool, SMSVSAM next checks the CFCACHE structure associated with the dataset. Again, if the record is located it is read into the local buffer pool and returned to CICS. If the record is not found in the CFCACHE structure, SMSVSAM performs an IO to DASD to locate the CI, registers an interest in the CI, puts the CI in the CF Structure and reads the records into its buffer pool.

(*) – If SMSVSAM on a different LPAR has updated the same CI then the version held in the local SMSVSAM buffer pools becomes invalid. In this case the new valid version must be re-read into the buffer pools from the coupling facility cache structure prior to the record being returned to the application.

The locks obtained and held during the read operation depend upon the READINTEG setting of the request. This is discussed in more detail later in the document.

EXEC CICS READ UPDATE

The processing for EXEC CICS READ UPDATE is essentially the same as those for read, with the exception that a record lock is requested and held until Syncpoint for files that are recoverable.

EXEC CICS REWRITE

In order to rewrite the record, SMSVSAM will have obtained an exclusive lock on the record in the IGWLOCK00 structure as part of the READ UPDATE.
When the rewrite request is issued the CI containing the updated record is updated in the local bufferpool followed by an update in both the CFCACHE structure and then DASD. During the write to DASD the Castout lock will be held on the CI.

Any SMSVSAM address space on a foreign LPAR will now have its copy of the CI that contains the updated record ‘cross invalidated’ and will need to refresh the bufferpool copy from the CFCACHE.

At Syncpoint the exclusive record lock is released.

Updates to different records in the same CI

If two CICS regions on differing LPARs within the Sysplex are updating different records in the same CI at the same time, the processing is as follows:

1) CICS1 on MVS1 reads record ‘A’ from CI ‘1’ at the same time as CICS2 on MVS2 reads record ‘B’ from CI ‘1’
2) CICS2 does its rewrite operation first. During the rewrite to DASD CICS2 holds a CASTOUT lock for the CI.
3) CICS1 tries to rewrite its version of the CI but is held up because it cannot get the CASTOUT lock held by CICS1. It will retry in 1.5 milliseconds to do the rewrite. If OA30499 is installed, this retry interval will be self tuning based on DASD response times.

4) When CICS1 does get to perform the rewrite it will ascertain that the CI has changed and the changes made by CICS1 must be merged back into the CI. This is performed 'under the covers' and is transparent to the application.
5) The re-applying of the record update to the CI is called a REDO as is seen in the SMF42 statistics data.

Setting up CICS to exploit RLS

The ability to access VSAM files using RLS is an integral part of the CICS TS product.
This means that any CICS application that is currently accessing VSAM files locally using LSR or by function shipping to a FOR region can access the same VSAM files in RLS mode. No modification or recompilation of the application program is required to affect this change. All that needs occur for this to happen is to alter the CICS file definition as described in CICS File Definitions below and to specify RLS=YES in the SIT.

CICS File definitions

The CICS resource definition for files is described in detail in the CICS Resource Definition Guide SC34-6815-00. When the file is opened, setting the RLSACCESS attribute to YES will result a CICS request to open the file in RLS mode.

Another attribute you should be aware of, as it can significantly affect performance, is READINTEG. The default setting of UNCOMMITTED will result in the same integrity as LSR access. Any other setting will result in an increase in locking activity for read requests.
You are advised to consult the CICS and DFSMS publications for a description of the affect each of these settings will have on READ requests.

CICS Program definitions

While attributes of the program have little effect on RLS file access there is one parameter that is worth mentioning with respect to RLS. This is CONCURRENCY.

The concurrency attribute determines whether the program runs under the QR TCB (when QUASIRENT is specified) or under an OTE TCB (when THREADSAFE is specified). In the first case, RLS file access requests are executed on dedicated SRBs while for the second case, the RLS access request is performed on OTE L8|L9 TCBs. For performance measurement purposes the SRB CPU usage is stored in the RLSCPUT field and is an accurate reflection of time consumed performing RLS activity. For the second case L8 CPU consumption is for all activity performed on the L8 and as such CPU used performing purely RLS processing cannot be determined.

Migration scenarios

To show the effect of migrating between configurations, we took one of our simple ‘in house’ benchmarks and ran it against various configurations.

The first configuration was a TOR/AOR where all the files were accessed locally in the AORs via LSR Pools. The next move was to introduce an FOR and function ship all file requests via MRO. The third scenario was to move the FOR to another LPAR in the Sysplex so that all file requests were function shipped across XCF via a Coupling Facility. The final scenario was to remove the FOR and redefine all of the files in the AORs as RLS. This causes all file requests to be performed by RLS via the SMSVSAM address space. We showed three flavours of RLS access in 3 separate benchmarks:

(1) **No Read Integrity (Uncommitted)**

No Lock accesses for Read. A Read will see records that might have been Read for Update by an application and are about to be changed with a Rewrite. This is the same as the situation for LSR files.

(2) **Consistent Read (CR)**

While a record has been Read for Update, no other transaction can Read or Browse the same record until it has been committed with a Syncpoint. A consistent Read has to obtain a shared lock, this cannot be done while someone else has an Exclusive Lock obtained as part of Read Update.

(3) **Repeatable Read (RR)**

While a record has been Read for Update, no other transaction can Read or Browse the same record until it has been committed. This is the same as Consistent Read, however when someone has Read a record, the Shared Lock that is obtained is held until the Reader has Syncpointed, hence no other transaction can Read for Update this record during this time.

Sysplex Configuration

The Sysplex consisted of 2 LPARs, each with 3 dedicated CPs on a 2097-763 Model E64. There were 2 Integrated CFs on the 2097 with 3 shared CPs each and Dynamic Dispatch turned off.

CICS Configurations

CICS TS V4.1 was used, running on zOS 1.11. Network simulators were used to drive the workload at approximately the same transaction rates for each scenario. All transactions entered the TOR and were routed to the AOR where file requests were either function shipped or accessed locally as LSR or RLS files.

Workload

The workload consisted of relatively small COBOL applications with little business logic. These applications were not defined as Threadsafe. The following are key attributes.

- Average of 6 file requests per transaction
- MRO Long running mirrors
- 69% Read, 10% Read for Update, 9% Update,11% Add , 1% Delete
- FCQRONLY=YES
- LOG(UNDO)

The RLS Buffer Pool size and CF caches etc. were sized so that we did not get any major performance constraints in the system. FCQRONLY=YES will give slightly better performance in a CICS region where you know that there are no threadsafe applications. If running in threadsafe mode this should be set to NO otherwise all file control requests will switch onto the QR TCB for processing.

Measurement methodology

Performance data was collected for 5 distinct transaction throughput rates. Network simulators entered transactions in the TORs using a particular ‘think time’. The workload was allowed time to settle and then performance data using RMF Monitor I, Monitor III, CICS Interval Statistics, CICS Performance Monitoring SMF 110s and RLS SMF 42 records was collected. The ‘think time’ was reduced and this process was repeated for 5 throughput rates. The tables below will only show the RMF I data for the workload, but examples of the other data collected is shown elsewhere in the report.

Understanding the following tables

The following tables show data that has been extracted from RMF Monitor I during the measurement intervals. The columns have the following meanings.

- **ETR - Transactions per second counted at the TOR**
- **TOR% - Percentage of 1 CP used by the TOR during the interval**
- **AOR% - Percentage of 1 CP used by the AOR during the interval**
- **FOR% - Percentage of 1 CP used by the FOR during the interval**
- **TOTAL CPU% - The total of the relevant address space CPU**
- **CPU/TRAN - Sum of address space CPU / Transaction rate**
- **RESP Time - Total transaction response time measured in the TOR**

Also shown when appropriate is the CPU% for XCFAS on both LPARs and SMSVSAM address space.

Scenario 1 (Local LSR access in the AOR)

ETR	TOR%	AOR%	TOTAL CPU%	CPU/TRAN	RESP Time
423.46	3.79	11.35	15.14	0.357ms	7ms
531.61	4.72	14.07	18.79	0.353ms	7ms
702.08	6.14	18.24	24.38	0.347ms	7ms
1041.40	9.03	27.04	36.07	0.346ms	7ms
1380.71	11.84	35.30	47.14	0.341ms	7ms

Figure 4. Cost of Local LSR access in the AOR

Using the data recorded above we can calculate the average CPU per transaction across the 5 intervals above to be 0.34ms.

Scenario 2 (MRO Function Shipping)

ETR	TOR%	AOR%	FOR%	TOTAL CPU%	CPU/TRAN	RESP Time
427.41	4.09	10.68	6.42	21.19	0.495ms	11ms
536.87	5.05	13.34	7.79	26.18	0.487ms	11ms
711.31	6.51	17.36	9.91	33.78	0.474ms	11ms
1065.26	9.52	25.57	14.67	49.76	0.467ms	11ms
1426.78	12.56	33.54	19.04	65.14	0.456ms	11ms

Figure 5. Cost of MRO function shipping

The average CPU per transaction after migrating to MRO function shipping has now gone up to 0.47ms. Because these applications have a low amount of business logic it would not be good practice to use this increase as a percentage increase but rather take the absolute deltas and apply these to the average of 6 Function ships per transaction. The delta increase is therefore 0.13ms/6 = 0.021ms per function ship. Now this number can be scaled from these 2097 703 cpu times to another processor using the LSPR and applied to other application profiles to get a rough estimate of the increase for your particular application.
This is the methodology that should be used for any of these measurement comparisons.

Scenario 3 (XCF Function Shipping)

ETR	TOR%	AOR%	XCFAS%	FOR%	XCFAS%	TOTAL CPU%	CPU/ TRAN	RESP Time
426.75	3.93	13.43	3.13	8.30	3.54	32.33	0.757	12ms
539.29	4.93	16.90	4.05	10.41	4.42	40.71	0.754	12ms
720.57	6.52	22.49	5.48	13.75	5.92	54.16	0.751	12ms
1072.64	9.69	33.42	8.17	20.04	8.92	80.24	0.748	12ms
1435.46	13.09	44.80	11.20	27.06	11.98	108.13	0.753	12ms

Figure 6. Cost of XCF function shipping

With the XCF function shipping scenario, we moved the FOR to another LPAR and then function shipped via the Coupling Facility. By doing this we also introduce CPU usage in the XCFAS address spaces on each LPAR. Now the average CPU per transaction has gone up to 0.75ms. Using the previous methodology you can calculate the cost per XCF function ship.

Scenario 4 (RLS Uncommitted Read)

ETR	TOR%	AOR%	SMSVSAM%	TOTAL CPU%	CPU/ TRAN	RESP Time
423.50	3.93	17.50	0.48	21.87	0.515	4ms
532.29	4.83	21.46	0.55	26.84	0.504	4ms
703.15	6.30	28.08	0.66	35.04	0.498	4ms
1042.72	9.12	40.91	0.90	50.93	0.488	4ms
1378.79	12.07	54.11	1.17	67.35	0.488	4ms

Figure 7. Cost of RLS uncommitted read

Notes on Read – Uncommitted

- This has the same characteristics as LSR files in that it is possible to read a record that has been Read for update by another task who might well change it with a Rewrite

In this scenario we removed the FOR from the configuration and defined all of the files in the AORs as RLS with uncommitted read integrity. If we were comparing this to the XCF function shipping then as the data shows, the CPU per transaction has now reduced to an average of 0.49ms.

Scenario 5 (RLS Consistent Read)

ETR	TOR%	AOR%	SMSVSAM%	TOTAL CPU%	CPU/ TRAN	RESP Time
423.64	4.00	20.63	0.50	25.13	0.593	5ms
532.09	5.00	25.49	0.56	31.05	0.583	5ms
703.11	6.46	33.00	0.69	40.15	0.571	5ms
1043.03	9.80	50.75	0.95	61.50	0.589	5ms
1378.54	12.32	64.26	1.22	77.80	0.564	5ms

Figure 8. Cost of RLS consistent read

Notes on Consistent Read

- This means that while a record has been Read for Update by a task, no other transaction can Read or Browse the same record until it has been committed with a Syncpoint.
- A consistent Read has to obtain a Shared Lock, this cannot be done while someone else has an Exclusive Lock obtained as part of Read Update.
- The Shared Lock is obtained and released as soon as the record is Read. This locking costs extra CPU and additional accesses to the Coupling Facility.

This is the same configuration as scenario 4 but the read integrity was changed to Consistent read. The average CPU per transaction is now 0.58ms.

Scenario 6 (RLS Repeatable Read)

ETR	TOR%	AOR%	SMSVSAM%	TOTAL CPU%	CPU/ TRAN	RESP Time
423.64	4.07	22.09	0.49	26.65	0.629	5ms
532.04	5.17	24.16	0.56	29.89	0.561	5ms
703.11	6.65	31.23	0.66	38.54	0.548	5ms
1043.03	9.49	45.17	0.92	55.58	0.532	5ms
1378.54	12.42	59.30	1.22	72.94	0.529	5ms

Figure 9. Cost of RLS repeatable read

Notes on Repeatable Read

- This means that while a record has been Read for Update, no other transaction can Read or Browse the same record until it has been committed. This is the same as Consistent, **however** when someone has Read a record, the Shared Lock that is obtained is held until the Reader has Syncpointed, hence no other transaction can Read for Update this record during this time.
- At the higher transaction rates, the CPU per transaction was slightly less than the Consistent Read case. This was mostly due to there being fewer accesses to the CF for Locks.
- When using Repeatable Read only the first Read in a transaction for a particular record will obtain a lock, multiple reads for the same record will not need multiple lock accesses so depending on access patterns, Repeatable Read could use less CPU per transaction but can restrict throughput.

Summary of Performance measurements

The results in the previous tables show that the best performance, in terms of CPU cost per Transaction, come from Local LSR files. As data is made available for sharing - using Function Shipping or RLS - the cost increases.

The net effect of migrating to RLS will depend upon the original configuration:

- If the migration is from MRO - with a high proportion of requests being function shipped across XCF links - then there could be a reduction in overall CPU cost per transaction.
- If the migration is from MRO/XM or Local files, then there will be an increase in CPU cost per transaction.

Our workload showed an approximate 5% increase in CPU cost per transaction when migrating from MRO/XM function shipping to RLS. Other workloads will vary depending on the path length of the application and the number of file requests per transaction.

RLS has better scaling capabilities than CICS Function Shipping because it isn't limited to a single file owning region that is constrained to the speed of a CP due to its single TCB architecture.

Monitoring the System

This section describes what data can be collected to help analyse the performance of CICS/RLS systems. It shows how to start collecting the data, how it can be post processed and talks about some of the key fields in the data.
When analysing the performance of CICS/RLS applications, data can be collected from the following:

CICS

- CICS Statistics - SMF Records 110 type 2
- CICS Monitoring – SMF Records 110 type 1

RMF Monitor I and Monitor III

- SMF Records 70-79
- Online interactive displays with Monitor III via RMFWDM

SMSVSAM

- SMF 42 Records Types 15, 16, 17 18, 19

The following descriptions of data collection are making the assumption that we just want to look at data from a few short intervals - at a problem time of the day - rather than activating CICS monitoring permanently.

CICS Data

CICS statistics

CICS Statistics can be used to determine the types of File accesses and the rates. They can also be used to determine the CPU usage on all TCBs.

Non-threadsafe applications will run on the QR TCB which will be constrained to the speed of a single CP. RLS requests in this mode run on a SRBs in the CICS address space.

Threadsafe applications run concurrently on L8/9 TCBs and under this condition the RLS request will also execute on the L8/9 TCB.

CICS statistics are only really useful when the interval they cover is relatively short and the counters are reset at the start of the interval.
This can be done in 2 ways:

(1) Use CEMT to Reset the statistics and then after the required time, use CEMT to PERFORM a Statistics collection.
(2) Use CEMT to set Interval Statistic recording to the required time interval. With this method you can set the END of day time so that you can synchronise collection with RMF and SMSVSAM recording.

When CICS Interval Statistics records are written the counters are automatically reset.

The following example was executed at 12:11. The interval has been set to 15mins and the end of day has been set to 12:15 so we will get a short Interval of 4 minutes and after that all 15 min intervals will be on the quarter hour (00, 15, 30 and 45).

```
CEMT INQ STAT
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Sta On         Int( 001500 ) End( 121500 )
Nex(121500)
```

The CICS statistics formatting program (DFHSTUP) can be used to format the CICS statistics records. Sample output from this job showing the File Access counts and types follows.

```
FILES - Requests Information
0  File     Get      Get Upd  Browse   Update   Add      Delete   Brws Upd VSAM EXCP Requests
   Name     Requests Requests Requests Requests Requests Requests Requests   Data     Index

   ACCUNTDB    12710        0        0        0        0        0        0    12710    12710
   ACCUNTDX    12425        0        0        0        0        0        0    12425    12425
   COMPOSDB    27864        0        0        0        0        0        0    27864    27864
   COMPOSDX    28566        0        0        0        0        0        0    28566    28566
   CUSTOMER     4707     1572        0     1572     6276        0        0    17093    25193
   CUSTOMEX     4644     1547        0     1547     6218        0        0    16985    24915
```

Figure 10. CICS statistics formatting program (DFHSTUP)

Notes on File Stats for RLS

For RLS, VSAM EXCP requests are a count of the number of calls to the system buffer manager, not as with LSR where it is the number of EXCPs. It includes calls that result in either a coupling facility cache access or an I/O

Below is an extract from the Dispatcher section from Statistics showing the total address space CPU since the last reset on TCBs and SRBs within the CICS address space.

```
Dispatcher Start Date and Time. . . . . : 08/27/2009
09:23:32.3348
Address Space CPU Time. . . . . :     00:00:28.624146
Address Space SRB Time. . . . . :     00:00:20.033857
```

Notice that in this case the SRB activity is relatively high, indicating that this application is not Threadsafe and so the RLS requests are not handled on Open TCBs but on SRBs.

CICS Monitoring

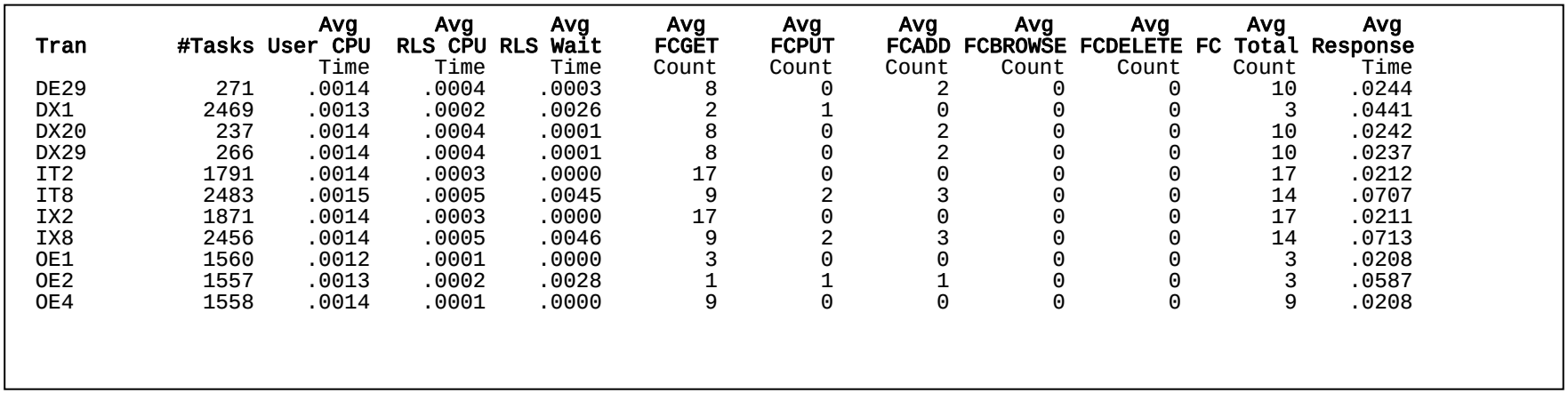
CICS Monitoring will generate an SMF 110 type 1 record for every transaction or if required, for long running transactions, at every Syncpoint. Monitoring can be set on via the SIT override MNPER=ON or dynamically using CEMT.
If you only want to collect for a 15 minute interval and coordinate it with CICS Statistics then use CEMT SET MON PERFCLASS(PERF) to turn it on at the start of the interval and NOPERF at the end of the interval to turn it back off. CICS Performance Monitoring can generate high numbers of SMF records and if permanently turned on will require processes to be in place to ensure correct sizing and offloading of the SMF datasets.

CICS Monitoring records can be processed using CICS Performance Analyser (PA). The following field names are of interest when looking at RLS performance.

CICS CMF name	CICS PA name	Description	CICS API requests
RLSCPUT	RLSCPU	RLS file request CPU (SRB) Time	CPU (SRB) time used to perform RLS request
RLSWAIT	RLSWAIT	File I/O wait	RLS file I/O wait time
FCGETCT	FCGET	File GET Requests	READ and READ UPDATE
FCPUTCT	FCPUT	File PUT Requests	REWRITE
FCBRWCT	FCBROWSE	File Browse Requests	READNEXT and READPREV
FCADDCT	FCADD	File ADD Requests	WRITE
FCDELCT	FCDELETE	File DELETE Requests	DELETE
FCTOTCT	FCTOTAL	Total file requests	Above plus STARTBR and ENDBR

Figure 11. CMF and PA field names for file accesses

CICS PA can generate many types of report. The following is an example of a Summary Report showing some of the data relevant to RLS applications.



Tran	#Tasks	Avg User CPU Time	Avg RLS CPU Time	Avg RLS Wait Time	Avg FCGET Count	Avg FCPUT Count	Avg FCADD Count	Avg FCBROWSE Count	Avg FCDELETE Count	Avg FC Total Count	Avg Response Time
DE29	271	.0014	.0004	.0003	8	0	2	0	0	10	.0244
DX1	2469	.0013	.0002	.0026	2	1	0	0	0	3	.0441
DX20	237	.0014	.0004	.0001	8	0	2	0	0	10	.0242
DX29	266	.0014	.0004	.0001	8	0	2	0	0	10	.0237
IT2	1791	.0014	.0003	.0000	17	0	0	0	0	17	.0212
IT8	2483	.0015	.0005	.0045	9	2	3	0	0	14	.0707
IX2	1871	.0014	.0003	.0000	17	0	0	0	0	17	.0211
IX8	2456	.0014	.0005	.0046	9	2	3	0	0	14	.0713
OE1	1560	.0012	.0001	.0000	3	0	0	0	0	3	.0208
OE2	1557	.0013	.0002	.0028	1	1	1	0	0	3	.0587
OE4	1558	.0014	.0001	.0000	9	0	0	0	0	9	.0208

Figure 12. CICS PA summary of file accesses

The above extract shows CPU time, Response time and file access counts. The RLS CPU time is the CPU time that RLS used to complete the request on the SRB. If this application was Threadsafe, all the application time and the RLS time would be accounted for on L8/9 TCBs and not SRBs. In this case RLS CPU would be reported as zero.

If you want to run the CICS PA File Summary reports that shows how many file accesses have been performed by transaction then you need MNRES=ON specified in the CICS SIT. **Note:** If you do not make changes to your MCT, by default it will only keep track of 8 files per transaction.
CICS standard monitoring performance class data includes totals for *all* files accessed by a transaction. Transaction resource monitoring, on the other hand, collects information about individual files, up to the number specified in the MCT. (DFHMCT TYPE= INITIAL,FILE=n)

CICS PA reports

Below are a number of CICS PA reports that show key aspects of the transactions

1) Performance total report – Summarising all activity performed by a transaction or transactions.
2) Wait analysis report – showing a breakdown of dispatch and wait information for the transaction/system.
3) Resource usage reports – two reports showing file access by transaction or by file.

For a detailed explanation of the all reports and features available consult the CICS Performance Analyser 3.1 documentation.

Performance total report

Statements to generate:

```
CICSPA IN(SMFIN001),
      APPLID(<applid>),
      LINECNT(60),
      FORMAT(':','/'),
      PRECISION(4),
      TOTAL(OUTPUT(<ddname>),
          TITLE1('<title>'),
          SELECT(PERFORMANCE(
          INC(TRAN(<transaction identifier>),
             START(FROM(,00:00:00.00),
                   TO(,23:59:59.99)),
              STOP(FROM(,00:00:00.00),
                   TO(,23:59:59.99))))))
```

The CICS Performance Analyser performance total report provides detailed statistics of all fields in the CMF performance class records.

```
                                  Dispatched Time          CPU Time
                                DD HH:MM:SS    Secs   DD HH:MM:SS    Secs

Total Elapsed Run Time             00:20:00    1200


From Selected Performance Records

QR Dispatch/CPU Time               00:00:08       8      00:00:04       4
MS Dispatch/CPU Time               00:00:00       0      00:00:00       0
                                   --------   -----      --------  ------
 TOTAL (QR + MS)                   00:00:08       8      00:00:04       4

L8 CPU Time                                               00:00:00       0
J8 CPU Time                                               00:00:00       0
S8 CPU Time                                               00:00:00       0
X8 CPU Time                                               00:00:00       0
                                   --------   -----      --------  ------
 TOTAL (L8 + J8 + S8 + X8)         00:00:00       0      00:00:00       0

L9 CPU Time                                               00:00:00       0
J9 CPU Time                                               00:00:00       0
X9 CPU Time                                               00:00:00       0
                                   --------   -----      --------  ------
 TOTAL (L9 + J9 + X9)              00:00:00       0      00:00:00       0

                                   --------   -----      --------  ------
```

Total CICS TCB Time	00:00:08	8	00:00:04	4

Total Performance Records (Type C)	0		
Total Performance Records (Type D)	0		
Total Performance Records (Type F)	0		
Total Performance Records (Type S)	0		
Total Performance Records (Type T)	302		
Total Performance Records (Selected)	302	Total Performance Records	20097

From Selected Performance Records	..... COUNTS ..... Total	Avg/Task	Max/Task	..... TIME ..... Total	Avg/Task	Max/Task
Dispatch Time	1823	6.0	235	8	.026	4.148
CPU Time				4	.012	1.899
RLS CPU (SRB) Time				3	.009	1.229
Suspend Time	1823	6.0	235	32	.104	11.152
Dispatch Wait Time	1521	5.0	234	5	.017	1.758
Dispatch Wait Time (QR Mode)	1521	5.0	234	5	.017	1.758
Response (-TCWait for Type C)				0	.000	.000
Response (All Selected Tasks)				39	.131	12.777
QR Dispatch Time	1823	6.0	235	8	.026	4.148
MS Dispatch Time	0	.0	0	0	.000	.000
RO Dispatch Time	0	.0	0	0	.000	.000
QR CPU Time				4	.012	1.899
MS CPU Time				0	.000	.000
RO CPU Time				0	.000	.000
L8 CPU Time				0	.000	.000
L9 CPU Time				0	.000	.000
J8 CPU Time				0	.000	.000
J9 CPU Time				0	.000	.000
S8 CPU Time				0	.000	.000
X8 CPU Time				0	.000	.000
X9 CPU Time				0	.000	.000

From Selected Performance Records	..... COUNTS ..... Total	Avg/Task	Max/Task	..... TIME ..... Total	Avg/Task	Max/Task
FCWAIT File I/O wait time	0	.0	0	0	.000	.000
RLSWAIT RLS File I/O wait time	715	2.4	40	16	.053	10.850
TSWAIT VSAM TS I/O wait time	0	.0	0	0	.000	.000
TSSHWAIT Asynchronous Shared TS wait time	125	.4	2	0	.001	.023
JCWAIT Journal I/O wait time	455	1.5	8	2	.007	.128
TDWAIT VSAM transient data I/O wait time	0	.0	0	0	.000	.000
IRWAIT MRO link wait time	3	.0	2	0	.000	.037
CFDTWAIT CF Data Table access requests wait time	0	.0	0	0	.000	.000
CFDTSYNC CF Data Table syncpoint wait time	0	.0	0	0	.000	.000
RUNTRWAI BTS run Process/Activity wait time	0	.0	0	0	.000	.000
SYNCDLY SYNCPOINT parent request wait time	0	.0	0	0	.000	.000
RMITIME Resource Manager Interface (RMI) elapsed time	1812	6.0	6	0	.000	.000
RMISUSP Resource Manager Interface (RMI) suspend time	0	.0	0	0	.000	.000
JVMITIME JVM initialize elapsed time	0	.0	0	0	.000	.000
JVMTIME JVM elapsed time	0	.0	0	0	.000	.000
JVMRTIME JVM reset elapsed time	0	.0	0	0	.000	.000
JVMSUSP JVM suspend time	0	.0	0	0	.000	.000
DB2CONWT DB2 Connection wait time	0	.0	0	0	.000	.000
DB2RDYQW DB2 Thread wait time	0	.0	0	0	.000	.000
DB2WAIT DB2 SQL/IFI wait time	0	.0	0	0	.000	.000
IMSWAIT IMS (DBCTL) wait time	0	.0	0	0	.000	.000
WMQGETWT WebSphere MQ GETWAIT wait time	0	.0	0	0	.000	.000

TCWAIT	Terminal wait for input time	0	.0	0	0	.000	.000
LU61WAIT	LU6.1 wait time	0	.0	0	0	.000	.000
LU62WAIT	LU6.2 wait time	0	.0	0	0	.000	.000
SZWAIT	FEPI services wait time	0	.0	0	0	.000	.000
SOWAIT	Inbound Socket I/O wait time	0	.0	0	0	.000	.000
OSOWAIT	Outbound Socket I/O Wait Time	0	.0	0	0	.000	.000
ISWAIT	IPCONN link wait time	0	.0	0	0	.000	.000
RQRWAIT	Request Receiver Wait Time	0	.0	0	0	.000	.000
RQPWAIT	Request Processor Wait Time	0	.0	0	0	.000	.000
DSPDELAY	First dispatch wait time	302	1.0	1	2	.008	.142
TCLDELAY	First dispatch TCLSNAME wait time	0	.0	0	0	.000	.000
MXTDELAY	First dispatch MXT wait time	0	.0	0	0	.000	.000
ENQDELAY	Local Enqueue wait time	0	.0	0	0	.000	.000
GNQDELAY	Global Enqueue wait time	0	.0	0	0	.000	.000

From Selected Performance Records		Total	Avg/Task	Max/Task	Total	Avg/Task	Max/Task
ICDELAY	Interval Control (IC) wait time	8	.0	6	9	.030	7.039
GIVEUPWT	Give up control wait time	212	.7	183	2	.005	1.423
WAITCICS	CICS ECB wait time	0	.0	0	0	.000	.000
WAITEXT	External ECB wait time	0	.0	0	0	.000	.000
PTPWAIT	3270 Bridge Partner wait time	0	.0	0	0	.000	.000
RRMSWAIT	Resource Recovery Services indoubt wait time	0	.0	0	0	.000	.000
LOCKDLAY	Lock Manager (LM) wait time	2	.0	2	0	.000	.010
DSTCBMWT	Dispatcher TCB Mismatch wait time	0	.0	0	0	.000	.000
MAXOTDLY	Maximum Open TCB delay time	0	.0	0	0	.000	.000
MAXJTDLY	Maximum JVM TCB delay time	0	.0	0	0	.000	.000
MAXSTDLY	Maximum SSL TCB delay time	0	.0	0	0	.000	.000
MAXXTDLY	Maximum XPLink TCB delay time	0	.0	0	0	.000	.000
DSMMSCWT	DS storage constraint wait time	0	.0	0	0	.000	.000
PCLOADTM	Program Library wait time	0	.0	0	0	.000	.000
SYNCTIME	SYNCPOINT processing time	302	1.0	1	1	.003	.125
OTSINDWT	OTS Indoubt Wait time	0	.0	0	0	.000	.000
EXWAIT	Exception Conditions wait time	0	.0	0	0	.000	.000
DSCHMDLY	Redispatch wait time caused by change-TCB mode	0	.0	0	0	.000	.000
TCMSGIN1	Messages received count	0	.0	0			
TCCHRIN1	Terminal characters received count	0	.0	0			
TCMSGOU1	Messages sent count	0	.0	0			
TCCHROU1	Terminal characters sent count	0	.0	0			
TCMSGIN2	Messages received from LU6.1	0	.0	0			
TCCHRIN2	LU6.1 characters received count	0	.0	0			
TCMSGOU2	Messages sent to LU6.1	0	.0	0			
TCCHROU2	LU6.1 characters sent count	0	.0	0			
TCALLOC	TCTTE ALLOCATE requests	0	.0	0			
TCM62IN2	LU6.2 messages received count	0	.0	0			
TCC62IN2	LU6.2 characters received count	0	.0	0			
TCM62OU2	LU6.2 messages sent count	0	.0	0			
TCC62OU2	LU6.2 characters sent count	0	.0	0			
ISALLOC	Allocate Session requests for sessions on IP	0	.0	0			
FCADD	File ADD requests	0	.0	0			
FCBROWSE	File Browse requests	895	3.0	192			
FCDELETE	File DELETE requests	4	.0	1			
FCGET	File GET requests	32798	108.6	18084			

From Selected Performance Records		Total	Avg/Task	Max/Task	Total	Avg/Task	Max/Task
FCPUT	File PUT requests	185	.6	4			
FCTOTAL	File Control requests	34013	112.6	18128			
FCAMCT	File access-method requests	34189	113.2	18129			
TDGET	Transient data GET requests	0	.0	0			
TDPUT	Transient data PUT requests	10	.0	10			
TDPURGE	Transient data PURGE requests	0	.0	0			
TDTOTAL	Transient data Total requests	10	.0	10			
TSGET	Temporary Storage GET requests	604	2.0	2			
TSPUTAUX	Auxiliary TS PUT requests	0	.0	0			

TSPUTMAI	Main TS PUT requests	0	.0	0
TSTOTAL	TS Total requests	604	2.0	2
BMSMAP	BMS MAP requests	187	.6	1
BMSIN	BMS IN requests	0	.0	0
BMSOUT	BMS OUT requests	252	.8	1
BMSTOTAL	BMS Total requests	439	1.5	2
JNLWRITE	Journal write requests	0	.0	0
LOGWRITE	Log Stream write requests	332	1.1	5
ICSTART	Interval Control START or INITIATE requests	0	.0	0
ICTOTAL	Interval Control requests	8	.0	6
SC24CGET	CDSA GETMAINs below 16MB	0	.0	0
SC31CGET	ECDSA GETMAINs above 16MB	7	.0	1
SC24CHWM	CDSA HWM below 16MB	0	.0	0
SC31CHWM	ECDSA HWM above 16MB	3440	11.4	512
SC24COCC	CDSA Storage Occupancy below 16MB	0	.0	0
SC31COCC	ECDSA Storage Occupancy above 16MB	0	.0	0
SC24UGET	UDSA GETMAINs below 16MB	32998	109.3	21716
SC31UGET	EUDSA GETMAINs above 16MB	136570	452.2	89975
SC24UHWM	UDSA HWM below 16MB	40783E3	135044.3	323360
SC31UHWM	EUDSA HWM above 16MB	20460E4	677486.5	1771920
SC24UOCC	UDSA Storage Occupancy below 16MB	4666	15.5	1943
SC31UOCC	EUDSA Storage Occupancy above 16MB	31378	103.9	15414
SC24SGET	CDSA/SDSA GETMAINs below 16MB	0	.0	0
SC24GSHR	CDSA/SDSA storage GETMAINed below 16MB	0	.0	0
SC24FSHR	CDSA/SDSA storage FREEMAINed below 16MB	0	.0	0
SC31SGET	ECDSA/ESDSA GETMAINs above 16MB	0	.0	0
SC31GSHR	ECDSA/ESDSA storage GETMAINed above 16MB	0	.0	0
SC31FSHR	ECDSA/ESDSA storage FREEMAINed above 16MB	0	.0	0
PCLINK	Program LINK requests	21064	69.7	11938
PCLOAD	Program LOAD requests	19138	63.4	11539
PCXCTL	Program XCTL requests	152	.5	2
PCLURM	Program LINK URM requests	303	1.0	2
PCDPL	Distributed Program Link (DPL) requests	0	.0	0
PCSTGHWM	Program Storage HWM above and below 16MB	22768E4	753920.6	2084304
PC24BHWM	Program Storage HWM below 16MB	235600	780.1	8680
PC31AHWM	Program Storage HWM above 16MB	22746E4	753170.8	2083808
PC24CHWM	Program Storage (CDSA) HWM below 16MB	0	.0	0
PC31CHWM	Program Storage (ECDSA) HWM above 16MB	1900808	6294.1	17856
PC24SHWM	Program Storage (SDSA) HWM below 16MB	186992	619.2	992
PC31SHWM	Program Storage (ESDSA) HWM above 16MB	19229E4	636736.4	1985536
PC24RHWM	Program Storage (RDSA) HWM below 16MB	52080	172.5	8680
PC31RHWM	Program Storage (ERDSA) HWM above 16MB	43081E3	142653.4	157376
DB2REQCT	DB2 requests	0	.0	0
IMSREQCT	IMS (DBCTL) requests	0	.0	0
WMQREQCT	Number of WebSphere MQ requests	0	.0	0
TCBATTCT	TCBs attached count	0	.0	0
DSTCBHWM	CICS Dispatcher TCB HWM	0	.0	0
CFCAPI	OO Foundation Class requests	0	.0	0
SYNCPT	SYNCPOINT requests	302	1.0	1
SOEXTRCT	EXTRACT TCP/IP and CERTIFICATE requests	0	.0	0
SOCNPSCT	Create Non-Persistent Outbound Socket reqs	0	.0	0
SOCPSCT	Create Persistent Outbound Socket requests	0	.0	0
SORCV	Outbound Sockets RECEIVE requests	0	.0	0
SOSEND	Outbound Sockets SEND requests	0	.0	0
SOTOTAL	Socket Total requests	0	.0	0
SOCHRIN	Outbound Sockets characters received count	0	.0	0
SOCHROUT	Outbound Sockets characters sent count	0	.0	0
SOMSGIN1	Inbound Sockets RECEIVE requests	0	.0	0

Wait analysis report

Statements to generate:

```
CICSPA IN(SMFIN001),
      APPLID(<applid>),
      LINECNT(60),
      FORMAT(':','/'),
      PRECISION(4),
      WAITANAL(OUTPUT(<ddname>),
          TITLE1('<title>'),
          SELECT(PERFORMANCE(
          INC(START(FROM(,00:00:00.00),
                TO(,23:59:59.99)),
             STOP(FROM(,00:00:00.00),
                TO(,23:59:59.99))))))
```

The CICS Performance Analyser wait analysis report provides a breakdown of wait activity by Transaction ID.

```
Tran=ABCD
  Summary Data                                          -------- Time ---------    ------ Count ------    ------ Ratio ------
                                                             Total     Average       Total    Average
    # Tasks                                                                            8
    Response Time                                         0.180056    0.022507
    Dispatch Time                                         0.070352    0.008794          34        4.3     39.1% of Response
    CPU Time                                              0.037625    0.004703          34        4.3     53.5% of Dispatch
    Suspend Wait Time                                     0.109703    0.013713          34        4.3     60.9% of Response
    Dispatch Wait Time                                    0.035403    0.004425          26        3.3     32.3% of Suspend
    Resource Manager Interface (RMI) elapsed time         0.000155    0.000019          48        6.0      0.1% of Response
    Resource Manager Interface (RMI) suspend time         0.000000    0.000000           0        0.0      0.0% of Suspend

  Suspend Detail                                        ------------------- Suspend Time ------------------- ----- Count -----
                                                             Total     Average   %age  Graph                  Total  Average
    JCIOWTT  Journal I/O wait time                        0.054932    0.006867  50.1% |**********               16      2.0
    RLSWAIT  RLS File I/O wait time                       0.039225    0.004903  35.8% |*******                  8      1.0
    DSPDELAY First dispatch wait time                     0.013380    0.001672  12.2% |**                       8      1.0
    TSSHWAIT Asynchronous Shared TS wait time             0.002165    0.000271   2.0% |                        2      0.3
```

Resource usage report

Statements to generate:

```
CICSPA IN(SMFIN001),
      APPLID(<applid>),
      LINECNT(60),
      FORMAT(':','/'),
      PRECISION(4),
      RESUSAGE(OUTPUT(<ddname>),
          FILESUMMARY(BYTRAN,TOTAL),
          TITLE1('<title>'),
          SELECT(PERFORMANCE(
          INC(START(FROM(,00:00:00.00),
                  TO(,23:59:59.99)),
              STOP(FROM(,00:00:00.00),
                  TO(,23:59:59.99))))))
```

The CICS Performance Analyser file usage summary provides a detailed analysis of CMF transaction resource class data for files.

The following report is a summary of files access per transaction.

```
                                        ********************* FC Calls ********************* ******** I/O Waits *******  AccMeth
Tran                  #Tasks              Get      Put     Browse    Add    Delete   Total     File      RLS     CFDT   Requests
----                 -------            -------- -------- -------- -------- -------- -------- -------- -------- -------- --------
ABC1                      37 Elapse Avg                                                        .0000    .0572    .0000
                             Max                                                             .0000    .2621    .0000
                          Count  Avg      236        2        2        3        0      248        0       10        0      255
                                 Max      388       14       28       10        1      410        0       21        0      425

                                        ********************* FC Calls ********************* ******** I/O Waits *******  AccMeth
Tran                  #Tasks              Get      Put     Browse    Add    Delete   Total     File      RLS     CFDT   Requests
----                 -------            -------- -------- -------- -------- -------- -------- -------- -------- -------- --------
ABC2                    7603 Elapse Avg                                                        .0000    .0124    .0000
                             Max                                                             .0000    .4372    .0000
                          Count  Avg       51        0        6        0        0       58        0        2        0       58
                                 Max      133        0     1225        0        0     1235        0       16        0     1235

FILE0001 Printed at 13:01:46 11/16/2009    Data from 11:12:42 11/16/2009 to 12:22:41 11/16/2009    APPLID CICSG1D1     Page       3
```

Resource usage report

Statements to generate:

CICSPA IN(SMFIN001),
  APPLID(<applid>),
  LINECNT(60),
  FORMAT(':','/'),
  PRECISION(4),
  RESUSAGE(OUTPUT(<ddname>),
    TRANLIST(FILE))

The following report is a summary of access by file for each transaction.

```
 V3R1M0                                                        CICS Performance Analyzer
                                                            Transaction Resource Usage List
                                                           ______________________________________________
RESU0001 Printed at 13:01:46 11/16/2009    Data from 11:12:42 11/16/2009                                                    Page    4188

                                        Request          Fcty  Conn                                        UOW R                 Response
Tran  Userid  SC TranType Term   LUName   Type   Program  T/Name Name      NETName        APPLID     Task Seq T   Stop Time      Time
---- -------- -- -------- ----  -------- ------- -------- ------ ---- ----------------- --------   ---- --- - ------------ --------
VSCA USR000A  TO UMD      <AJ0  CICSG1A1 FS:F--- LGACVS01 T/<AJ0 G1A1 GBIBMIYA.GNWS008  CICSG1D1  61344   1 T 12:22:29.122    .0167

                                ********************** FC Calls ********************* ******* I/O Waits ********  AccMeth
      File                        Get      Put     Browse     Add     Delete    Total     File     RLS      CFDT   Requests
     --------                   -------- -------- -------- -------- -------- -------- -------- -------- -------- --------
     KSDSCUST           Elapse    .0000    .0000    .0000    .0038    .0000    .0038    .0000    .0000    .0000
                        Count         0        0        0        1        0        1        0        0        0        2

     *CTLACB*           Elapse    .0000    .0000    .0000    .0000    .0000    .0000    .0000    .0000    .0000
                        Count         0        0        0        0        0        0        0        0        0        1

     Total              Elapse    .0000    .0000    .0000    .0130    .0000    .0130    .0000    .0000    .0000
                        Count         0        0        0        3        0        3        0        0        0        9
```

RMF Monitor I

RMF Monitor I is a started task that writes SMF records 71 to 79 on a time interval basis. These records contain overall system performance data. ERBRMFPP can be used to post process these records. For this study the main area of interest was CICS transaction counts and CICS CPU usage.
There is more than one way to achieve this but for our benchmarks we put CICS regions into unique reporting groups using WLM. In the Classification rules for a given CICS region, set both the entry for 'CICS Transaction classification' and the 'Rules for JES batch' to the same Report Group.
In the following example Jobname CICS2A01 is a CICS region running with Applid IYCUZC01.

From the **CICS Transaction Classification**:

	--------Qualifier--------			-------Class--------	
Action	Type	Name	Start	Service	Report
			DEFAULTS:	CICSDEF	CICS
____	1 SI	IYCUZC06	___	CICSTRAN	ZC06
____	1 SI	IYCUZC25	___	CICSTRAN	CICS2A25
____	1 SI	IYCUZC10	___	CICSTRAN	ZC10
____	1 SI	**IYCUZC01**	___	CICSTRAN	**CICS2A01**

And from the **JES Batch Classification**:

	--------Qualifier--------			-------Class--------	
Action	Type	Name	Start	Service	Report
			DEFAULTS:	BATCH	BATCH
____	1 TN	**CICS2A01**	___	CICSBTCH	**CICS2A01**
____	1 TN	CICS2A10	___	CICSBTCH	CICS2A10
____	1 TN	CICS2A25	___	CICSBTCH	CICS2A25

Figure 13. RMF monitor I reports

To post process the RMF I SMF records use the following parameters in ERBRMFPP.

```
//SYSIN    DD *
 NOSUMMARY
```

SYSOUT(A)
REPORTS(ALL)
SYSRPTS(WLMGL(POLICY,WGPER,RCLASS,SCPER))

The following is an extract from the report produced.

```
-TRANSACTIONS-  TRANS-TIME HHH.MM.SS.TTT  --DASD I/O--  ---SERVICE---   SERVICE TIME  ---APPL %---
   AVG       1.00  ACTUAL                51  SSCHRT  1793  IOC      4281K  CPU  107.977  CP     28.40
   MPL       1.00  EXECUTION             18  RESP     0.2  CPU     34622K  SRB   17.835  AAPCP   0.00
   ENDED   495375  QUEUED                 0  CONN     0.1  MSO     24057M  RCT    0.000  IIPCP   0.00
   END/S  1085.80  R/S AFFIN              0  DISC     0.0  SRB      5719K  IIT    3.808
   #SWAPS       4  INELIGIBLE             0  Q+PEND   0.0  TOT     24102M  HST    0.000  AAP     0.01
   EXCTD        0  CONVERSION             0  IOSQ     0.0  /SEC    52829K  AAP    0.044  IIP      N/A
```

In the above report we see that in this run the CICS region is executing 1085.8 transactions per second with an average response time of 51ms while using 28.40% of 1 CP.

RMF Monitor III

RMF Monitor III can be started in addition to the Monitor I to monitor the Coupling Facility performance. Monitor III will also write SMF records which can be post processed using ERBRMFPP or it has an interactive monitor (RMFWDM) that you can use with TSO. Using the interactive monitor you can view Coupling Facility performance and RLS bufferpool activity.
To post process the SMF records to view CF and individual CF Cache performance use the following parameters in ERBRMFPP

```
//SYSIN    DD *
  NOSUMMARY
  SYSRPTS(CF)
  SYSOUT(A)
```

This will produce a report of which there are some extracts below.

```
STRUCTURE SUMMARY
    ------------------------------------------------------------------------------------------------------------------------

                                                  % OF                 % OF    % OF     AVG     LST/DIR  DATA       LOCK     DIR REC/
          STRUCTURE                     ALLOC     CF         #         ALL     CF       REQ/    ENTRIES  ELEMENTS  ENTRIES  DIR REC
    TYPE  NAME              STATUS CHG  SIZE      STOR       REQ       REQ     UTIL     SEC     TOT/CUR  TOT/CUR   TOT/CUR  XI'S

    LOCK  IGWLOCK00         ACTIVE        4M       0.4      4393K     85.3    74.0     14546    4739        0        524K      N/A
                                                                                                  131        0        674       N/A

    CACHE CACHE01           ACTIVE      245M      26.3     406074      7.9    14.0     1344.6     57K      113K      N/A         0
                                                                                                   56K      112K      N/A         0
          CACHE02           ACTIVE      245M      26.3     350582      6.8    11.6     1160.9     57K      113K      N/A         0
                                                                                                   56K      112K      N/A         0
```

Figure 14RMF monitor III reports

The above shows the Structure Summary. In this case it is giving information about the Lock structure and 2 Cache Structures that are being used for RLS files.

Notes on the Structure summary

- Look out for the DIR REC, these are an indication that the Structure is too small
- Also a high total number of DIR Entries compared to number of Data Elements indicate Elements are being reclaimed in favour of DIR Entries and that the structure is too small.

When the Structure is first allocated the ratio is defined by SMSVSAM and as the time goes by, this will dynamically adjust, depending on the CI size etc. The smaller the CI size the less Data Entries it takes to hold the data. If the demand for storage in this structure increases then

Entries will be reclaimed on an LRU algorithm. The algorithm will favour Directory Entries and reclaim Data Entries first. This is because, if a Data Entry is reclaimed it does not need to cross invalidate any of the local buffers because they are still valid. If a Directory Entry is reclaimed, then the control information about the buffer that it is related to is lost and therefore data integrity would be lost if all the MVS images who had registered an interest in this buffer did not have their local copies invalidated at this time.
The fact that there are more Directory Entries than Data Entries shows the favouring of Directory Entries and the dynamic adjustment trying to prevent Directory reclaims by increasing the proportion of them.

```
STRUCTURE NAME = IGWLOCK00          TYPE = LOCK   STATUS = ACTIVE
          # REQ    ------------- REQUESTS -------------   -------------- DELAYED REQUESTS -------------
SYSTEM    TOTAL             #     % OF  -SERV TIME(MIC)-    REASON    #   % OF  ---- AVG TIME(MIC) -----  EXTERNAL REQUEST
NAME      AVG/SEC           REQ    ALL     AVG   STD_DEV            REQ    REQ   /DEL    STD_DEV   /ALL   CONTENTIONS

MV2A       4393K   SYNC   4393K    100     3.1      1.4     NO SCH    0    0.0    0.0        0.0    0.0   REQ TOTAL     4396K
          14546    ASYNC     0     0.0     0.0      0.0     PR WT     0    0.0    0.0        0.0    0.0   REQ DEFERRED   212
                   CHNGD     0     0.0  INCLUDED IN ASYNC   PR CMP    0    0.0    0.0        0.0    0.0   -CONT          212
                                                                                                         -FALSE CONT    211

-------------------------------------------------------------------------------------------------------------------------------
TOTAL      4393K   SYNC   4393K    100     3.1      1.4     NO SCH    0    0.0    0.0        0.0    0.0   REQ TOTAL     4396K
          14546    ASYNC     0     0.0     0.0      0.0     PR WT     0    0.0    0.0        0.0    0.0   REQ DEFERRED   212
                   CHNGD     0     0.0                      PR CMP    0    0.0    0.0        0.0    0.0   -CONT          212
                                                                                                         -FALSE CONT    211
```

```
STRUCTURE NAME = CACHE01            TYPE = CACHE  STATUS = ACTIVE
          # REQ    ------------- REQUESTS -------------   -------------- DELAYED REQUESTS -------------
SYSTEM    TOTAL             #     % OF  -SERV TIME(MIC)-    REASON    #   % OF  ---- AVG TIME(MIC) -----
NAME      AVG/SEC           REQ    ALL     AVG   STD_DEV            REQ    REQ   /DEL    STD_DEV   /ALL

MV2A        406K   SYNC    406K    100     6.1      2.2     NO SCH    0    0.0    0.0        0.0    0.0
           1345    ASYNC     0     0.0     0.0      0.0     PR WT     0    0.0    0.0        0.0    0.0
                   CHNGD     0     0.0  INCLUDED IN ASYNC   PR CMP    0    0.0    0.0        0.0    0.0
                                                            DUMP      0    0.0    0.0        0.0    0.0

-------------------------------------------------------------------------------------------------------------------------------
TOTAL       406K   SYNC    406K    100     6.1      2.2     NO SCH    0    0.0    0.0        0.0    0.0    -- DATA ACCESS ---
           1345    ASYNC     0     0.0     0.0      0.0     PR WT     0    0.0    0.0        0.0    0.0    READS        41410
                   CHNGD     0     0.0                      PR CMP    0    0.0    0.0        0.0    0.0    WRITES      181944
                                                            DUMP      0    0.0    0.0        0.0    0.0    CASTOUTS         0
                                                                                                          XI's           500
```

The previous 2 reports show data about the individual structures in more detail.
Notes on the Cache report.

- **READS** are those accesses that are satisfied from the CF after not being found in the Bufferpool
- **WRITES** include cases where the record was not found in the Bufferpool or the CF but were found on DASD. The record is then put in the Bufferpool and written back to the CF. Application WRITES and REWRITES are also recorded here.

- **CASTOUTS** are not relevant to RLS. The RLS caching design writes "unchanged" entries to the cache. This means that the data in the cache matches the data on DASD. When unchanged entries are written, the CF can reclaim the entries as necessary for space (since the data is out on DASD and can be retrieved when required). Some XCF exploiters (like DB2 for instance), write "changed" entries. With changed entries, the CF is not allowed to reclaim them and the exploiters must then CASTOUT the entries to free up space. So with the RLS design, you will never see any CASTOUTs.
- **XI's** are Cross Invalidates. This is case where a CI resides in a the local SMSVSAM Bufferpool but a user on another LPAR has updated a record in it so the CI is marked as invalid and must be refreshed from the CF. XI's can also happen if the Cache size is too small and Directory Entries get reclaimed. Buffers associated with these need to be marked as XI. This was explained earlier under the title 'Notes on Structure Summary'.

RMF III Online Interactive

On TSO, if you use CLIST RMFWDM and choose option S for Sysplex views, you get the following panel which gives options for viewing the current activity in RLS.

```
Sysplex Reports
         1 SYSSUM   Sysplex performance summary                (SUM)
         2 SYSRTD   Response time distribution                (RTD)
         3 SYSWKM   Work Manager delays                       (WKM)
         4 SYSENQ   Sysplex-wide Enqueue delays                (ES)
         5 CFOVER   Coupling Facility overview                 (CO)
         6 CFSYS    Coupling Facility systems                  (CS)
         7 CFACT    Coupling Facility activity                 (CA)
         8 CACHSUM  Cache summary                             (CAS)
         9 CACHDET  Cache detail                              (CAD)
        10 RLSSC    VSAM RLS activity by storage class        (RLS)
        11 RLSDS    VSAM RLS activity by data set             (RLD)
        12 RLSLRU   VSAM LRU overview                         (RLL)
```

Figure 15. RMF Monitor III online interactive options

Above is the main Sysplex panel and from here you can see detail on Coupling Facility performance, individual Cache activity, RLS activity by Storage Class/Dataset and RLS Bufferpool activity.

SMSVSAM Monitoring

SMSVSAM writes SMF 42 records on an interval basis. While data for the SMF 42 records is collected by each SMSVSAM address space running within the SYSPLEX it is only written by one designated SMSVSAM address space. To determine upon which LPAR the data is being recorded issue the ***D SMS,SMSVSAM,ALL*** command. The SMF 42 status information will contain an * next to the SYSNAME where recording is taking place. In the display below this is MVA0.

```
DISPLAY SMS,SMSVSAM - SMF RECORD 42 STATUS
                    SMF_TIME  CF_TIME ----- SUB-TYPE  SUMMARY -----
                                          15  16  17  18  19
  SYSNAME: MVA2      YES- 4    3600- 4     YES YES YES YES YES
  SYSNAME: MVA3      YES- 3    3600- 3     YES YES YES YES YES
  SYSNAME: MVA0     *YES- 1    3600- 1     YES YES YES YES YES
  SYSNAME: MVA1      YES- 2    3600- 2     YES YES YES YES YES
  SYSNAME: ........ ....-..  ......-..     ... ... ... ... ...
  SYSNAME: ........ ....-..  ......-..     ... ... ... ... ...
  SYSNAME: ........ ....-..  ......-..     ... ... ... ... ...
  SYSNAME: ........ ....-..  ......-..     ... ... ... ... ...
```

Figure 16. D SMS,SMSVSAM,ALL command output

The following SMS parameters affect the frequency and synchronisation of SMF 42 records:

- CF_TIME – Aligns creation of all the CF related SMF 42 records with subtype 15,16,17,18 and 19 and dictates the interval at which they are written.
- SMF_TIME – Aligns SMF type 42 records for DFSMS (subtype 1,2,15,16,17,18 and 19) to the SMF_TIME interval

Note: SMF_TIME overrides the value of CF_TIME (plus BMFTIME and CACHETIME) so you can synchronise the SMF 42 records with your measurement time. CF_TIME can be changed dynamically using **SETSMS CF_TIME(nn)** where **nn** is the length of the interval in minutes.

DFHSMS generates SMF type 42 records with subtype 15-19 being relevant to RLS as follows:

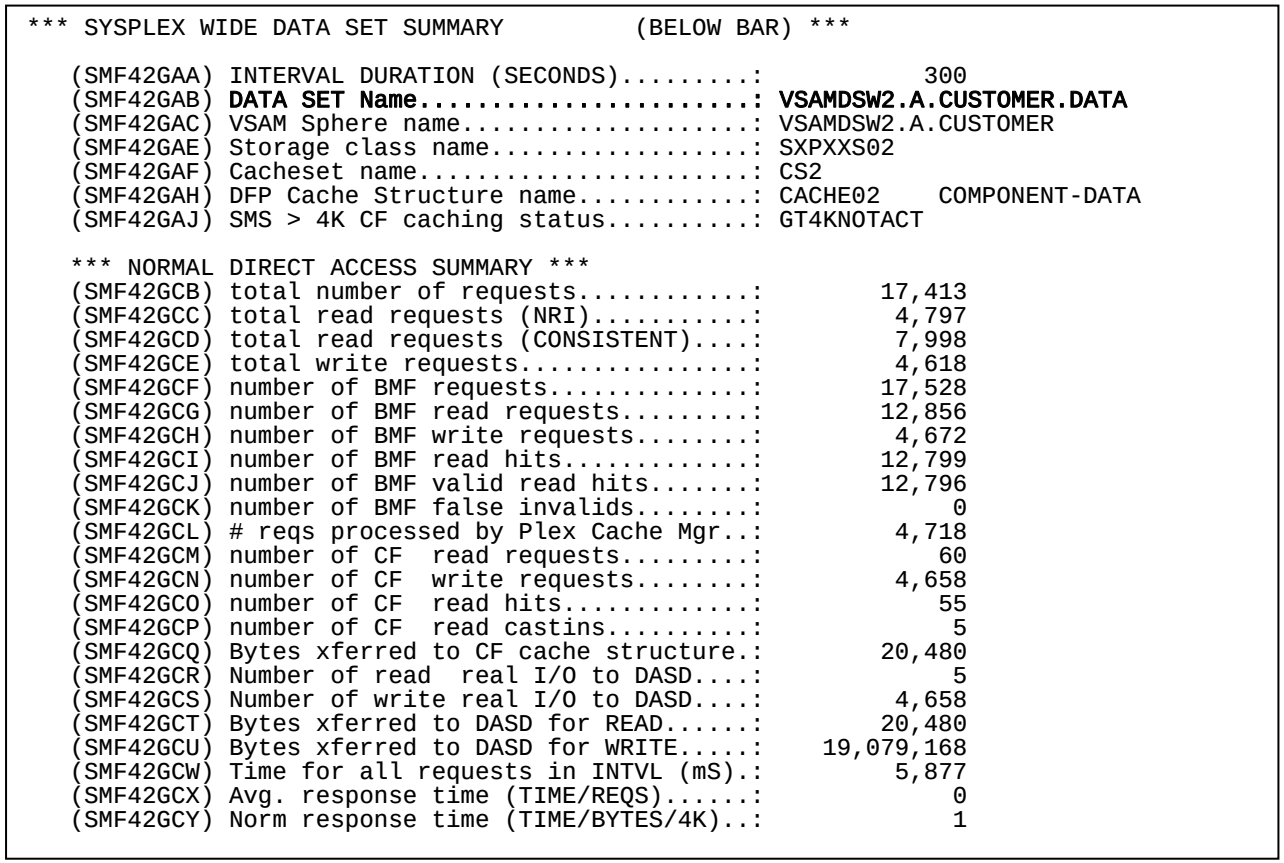
- Subtype 15 – VSAM RLS Storage Class Response Time Summary
- Subtype 16 – VSAM RLS Dataset Response Time Summary (*)
- Subtype 17 – VSAM RLS Coupling Facility Lock Structure Usage

- Subtype 18 – VSAM RLS CF Cache Partition Usage
- Subtype 19 – VSAM RLS Local Buffer Manager LRU statistics summary

(*) Generation of the subtype 16 records is controlled using the **V SMS,MONDS** command.
The SMF 42 type records, by default collect data associated with Storage Classes. As Storage Classes have numerous Datasets allocated within them, this can prevent analysis of individual Files. This should be considered when planning the allocation of datasets within Storage Classes.
Monitoring of individual datasets can be achieved by using **V SMS,MONDS(xxx.yyy.www),ON** where xxx.yyy.www is a dataset name.

SMF 42 Type 16 records

To give an idea of the performance data available from SMSVSAM, the following report extracts are at the dataset level from SMF 42 type 16 records.
The next 3 report extracts show performance data with regards to the Data component of dataset VSAMDSW2.A.CUSTOMER
The first shows the types and numbers of accesses to the bufferpool, the Coupling Facility and to DASD.
The second shows the information about the locks for this component.
The third shows information about the bufferpool. The SMF record number is shown in the report and can be used to reference the description.
These 3 extracts are also generated for the Index component of this dataset.
The extracts are also repeated for the SYSPLEX view and for all LPAR views.



```
*** SYSPLEX WIDE DATA SET SUMMARY         (BELOW BAR) ***

   (SMF42GAA) INTERVAL DURATION (SECONDS).........:           300
   (SMF42GAB) DATA SET Name.....................: VSAMDSW2.A.CUSTOMER.DATA
   (SMF42GAC) VSAM Sphere name..................: VSAMDSW2.A.CUSTOMER
   (SMF42GAE) Storage class name................: SXPXXS02
   (SMF42GAF) Cacheset name.....................: CS2
   (SMF42GAH) DFP Cache Structure name..........: CACHE02    COMPONENT-DATA
   (SMF42GAJ) SMS > 4K CF caching status........: GT4KNOTACT

   *** NORMAL DIRECT ACCESS SUMMARY ***
   (SMF42GCB) total number of requests..........:        17,413
   (SMF42GCC) total read requests (NRI).........:         4,797
   (SMF42GCD) total read requests (CONSISTENT)..:         7,998
   (SMF42GCE) total write requests..............:         4,618
   (SMF42GCF) number of BMF requests............:        17,528
   (SMF42GCG) number of BMF read requests.......:        12,856
   (SMF42GCH) number of BMF write requests......:         4,672
   (SMF42GCI) number of BMF read hits...........:        12,799
   (SMF42GCJ) number of BMF valid read hits.....:        12,796
   (SMF42GCK) number of BMF false invalids......:             0
   (SMF42GCL) # reqs processed by Plex Cache Mgr..:         4,718
   (SMF42GCM) number of CF  read requests.......:            60
   (SMF42GCN) number of CF  write requests......:         4,658
   (SMF42GCO) number of CF  read hits...........:            55
   (SMF42GCP) number of CF  read castins........:             5
   (SMF42GCQ) Bytes xferred to CF cache structure.:       20,480
   (SMF42GCR) Number of read  real I/O to DASD..:             5
   (SMF42GCS) Number of write real I/O to DASD..:         4,658
   (SMF42GCT) Bytes xferred to DASD for READ....:       20,480
   (SMF42GCU) Bytes xferred to DASD for WRITE...:    19,079,168
   (SMF42GCW) Time for all requests in INTVL (mS).:       5,877
   (SMF42GCX) Avg. response time (TIME/REQS)....:             0
   (SMF42GCY) Norm response time (TIME/BYTES/4K)..:           1
```

Figure 17. SMF 42 formatting program output

```
*** LOCK STATISTICS SUMMARY ***
   (SMF42GPA) # of record lock reqs (OBT/ALT/PROM):         8,018
   (SMF42GPB) # reqs with true lock contention....:             0
   (SMF42GPC) # reqs with false lock contention...:             0
   (SMF42GPD) # of record lock release requests...:         7,962
   (SMF42GPE) # of component_1 locks requests.....:            29
   (SMF42GPF) # of component_1 release requests...:             0
   (SMF42GPH) # of comp_1, class_1, DIWA locks....:            29
   (SMF42GPI) # DIWA locks - true contention......:             0
   (SMF42GPJ) # DIWA locks - false contention.....:             0
   (SMF42GPK) # DIWA locks - release locks........:             0
   (SMF42GPL) # of comp_1, class_2, UPGRADE locks.:             0
   (SMF42GPM) # UPGRADE locks - true contention...:             0
   (SMF42GPN) # UPGRADE locks - false contention..:             0
   (SMF42GPO) # UPGRADE locks - release locks.....:             0
   (SMF42GPP) # of comp_1, class_3,PREFORMAT locks:             0
   (SMF42GPQ) # PREFORMAT locks - true contention.:             0
   (SMF42GPR) # PREFORMAT locks - false contention:             0
   (SMF42GPS) # PREFORMAT locks - release locks...:             0
   (SMF42GPT) # of component_2 locks requests.....:             0
   (SMF42GPU) # Comp_2 locks - true contention....:             0
   (SMF42GPV) # Comp_2 locks - false contention...:             0
   (SMF42GPW) # Comp_2 locks - release locks......:             0
```

Notes on Locking

- Locking of individual records within VSAM data sets is a central element of the overall design. RLS also uses other higher-level locks for serializing operations such as control interval and control area split.
- DIWA Locks (Data Insert Work Area) are for Splits and ESDS Add to End
- PREFORMAT is when a new CA has been acquired
- UPGRADE is for AIX support

```
*** DATASET BMF LRU STATISTICS ***
   (SMF42GRA) Number of REDOs.....................:            14
   (SMF42GRB) Number of Recursive REDOs...........:             0
   (SMF42GRC) Number of BMF writes................:         4,619
   (SMF42GRD) Number of SCM reads.................:            60
   (SMF42GRE) Number of SCM reads (castout lock)..:             7
   (SMF42GRG) Percentage REDOs....................:             0
   (SMF42GRH) Percentage Recursive REDOs..........:             0
   (SMF42GRI) Percentage SCM CASTOUT LOCK.........:             3
```

Notes on BMF LRU Statistics

- RLS uses the buffer registration and invalidation functions of the coupling facility cache as the means to maintain buffer coherency across the local buffer pools
- There are multiple copies of a CI, transactions have their own copy and each may be updating a different record within it.
- The CASTOUT Lock is held while a CI is written to DASD by the first user to rewrite the CI
- This prevents multiple users writing changes to the same CI at the same time
- The subsequent ones fail when they attempt to write and the copy they have is merged with the new one. This is called a REDO and is handled under the covers without the knowledge of the application program.
- The redo is retried every 1.5ms while the Castout lock is held
- It is therefore possible to get recursive REDOs
- SCM reads (Castout lock) is the count of the times someone tried to Read a CI but the Castout lock was held.

Relative Path length numbers

The following table shows the costs in micro seconds on a z10 for various CICS Local LSR and RLS API calls.
The first row shows a cost of 5 micro-seconds when the record is found in the LSR buffer pool and 16 micro-seconds when it has to go to DASD. With RLS, if the record is not found in the Bufferpool, it goes to the CF, if not found there, it then goes to DASD. RLS then puts the CI in the Bufferpool and also writes it to the CF.

These figures are for guidance only and may vary significantly depending on many situations at the time of the request

	BUFF LSR	DASD LSR	BUFF RLS	CACHE RLS	DASD RLS
Read	**5**	**16**	**17**	**33**	**55**
Read/upd	**9**		**55**		
Rewrite		**24**			**60**
Write		**25**			**90**
Delete	**18**		**91**		

Useful Commands

D SMS,SMSVSAM,ALL
Show prime SMSVSAM server in Sysplex (the SMF writer)
D SMS,CFCACHE(*)
Show status of RLS CF Cache Structures
D SMS,CFLS
Show Lock rates and contention on Lock structure
V SMS,MONDS('dsname'),ON
Start Dataset monitoring
D SMS,MONDS(*)
Show which Datasets are being monitored
SETSMS RLS_MAX_POOL_SIZE(nn)
Change the size of the RLS Bufferpool
SETSMS RLSABOVETHEBARMAXPOOLSIZE(sysid,nn)
Change the size of the RLS Bufferpool above the Bar
SETSMS RLSFIXEDPOOLSIZE(sysid,nn)
Allocate fixed Real storage to the bufferpool
SETSMS CF_TIME(600)
Change the interval for recording SMSVSAM performance data
V SMS,CFCACHE(CACHE02),ENABLE
Enable a CF Cache for RLS usage
D SMS,OPTIONS
Show all the current SMSVSAM options
D SMF,O
Show the current SMF settings
SETXCF START,ALTER,STRNM='strname',SIZE=nnn
Change the size of a structure dynamically
ISMF
Provides detailed Storage Management information

RLS Terminology

Record lock
This is an XES lock resource obtained by SMSVSAM on behalf of a user and associated with a logical record. The lock resource name is based on a 16 byte hashed version of the record's key (or RBA, or RRN), as well as the sphere name and component name. There are also other locks to serialize splits, for example.

True contention
This is when two different "users" attempt to access the same record lock at the same time. Reported as true contention on the D SMS,CFLS command and in RMF reports. You need to tune your application if you have high true contention rates, or use NRI for reads.

False contention
This is when VSAM RLS assigns locked resources to an entry value in the lock table, and uses this entry value to quickly check whether a resource is already locked. If the lock structure (and thus the lock table) is too small, many locks can be represented by a single value, making "false" lock contention possible. False lock contention occurs when two different locks on different resources attempt to use the same lock entry. The second lock requester is suspended until VSAM RLS determines that there is no real lock contention on the resource.
This is reported as false contention on the D SMS,CFLS command. For high false contention rates need to create a larger lock structure.

True/False or False/False contention
This occurs when either true or false contention exists, and the holder releases the record before the RLS contention exit runs. At this point RLS cannot tell between true or false contention. This type of contention is reported as false contention on the D SMS,CFLS command.

Deadlocks
This occurs when two different transactions, each holding a record lock, attempt to obtain the other transactions record lock. SMSVSAM abnormally terminates one of the waiting lock requests (RPL RC=8 RSN=21), based on the Deadlock_detection value in your IGDSMSxx.

Read integrity

SMSVSAM allows the user to decide about read integrity through parameters in DD card or ACB macro. If the option is consistent read (CR), logical records locks are required in shared mode for reads. If the option is no read integrity (NRI), no locks are required along reads.

Retained lock
This is a record lock obtained by a recoverable subsystem such as CICS, which was held at the time of a failure in: CICS, IGWLOCK00, CF, z/OS, other SMSVSAM, or Indoubt transactions. A job can be cancelled without retained locks because this job does not register, as subsystem with SMSVSAM.

Lost Locks

A sphere is said to be in "Lost Locks" if the sphere was being accessed by a recoverable subsystem when a failure to the lock structure occurs at the same time as a failure in at least one of the SMSVSAM address spaces (double failure scenario). Use the IDCAMS SHCDS LISTSUBSYS(ALL) command to list CICS subsystems holding retained or lost locks. This situation can be avoided by duplexing or failure isolation.
CEMT INQ DSNAME can also give additional information about datasets in lost lock status.

Reference documentation

Redbook: VSAM Demystified
CICS and VSAM Record Level Sharing: Recovery Considerations, SG24-4768-00
CICS and VSAM Record Level Sharing: Implementation Guide, SG24-4766-00
CICS and VSAM Record Level Sharing: Planning guide, SG24-4765-00

SMF 42 formatter reports

The supplied object library contains object code for two modules:

- SMF42PRC
- TIMECONV

JCL to link edit

```
//LKED   EXEC PGM=HEWL,PARM='LIST,XREF,RENT'
//SYSLIB   DD DISP=SHR,DSN=<hlq>.SCEELKED
//SYSLMOD  DD DISP=SHR,DSN=<your load library>
//OBJLIB   DD DISP=SHR,DSN=<supplied objlib>
//SYSUT1   DD UNIT=SYSDA,DCB=BLKSIZE=1024,
//            SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=<supplied objlib>(SMF42PRC)
//         DD DDNAME=SYSIN
//SYSIN    DD *
  INCLUDE OBJLIB(TIMECONV)
  ENTRY SMF42PRC
  NAME SMF42PRC(R)
/*
```

JCL to execute

```
//S001  EXEC PGM=SMF42PRC,REGION=0M
//STEPLIB  DD DISP=SHR,DSN=<your load library>
//         DD DISP=SHR,DSN=<hlq>.SCEERUN
//SUMMARY  DD SYSOUT=*
//SUBTYP15 DD SYSOUT=*
//SUBTYP16 DD SYSOUT=*
//SUBTYP17 DD SYSOUT=*
//SUBTYP18 DD SYSOUT=*
//SUBTYP19 DD SYSOUT=*
//SMFIN    DD DISP=SHR,DSN=<input SMF dataset>
```

Sample output

The SMFPRC42 program should terminate with a completion code of 0.

The SUMMARY output DD will contain an overview of the processing which includes a counts of SMF records read, SMF 42 records found including a count of each subtype record processed.

Output for a particular subtype can be suppressed by removing the corresponding DD. For instance, to suppress output for subtype 16 remove the SUBTYP16 DD from the JCL.

Ignore the message concerning filtering. This version of the program does not support filtering.

Each SUBTYPxx DD will contain a formatted version of the SMF 42 subtype found. All data fields in the SMF record are formatted with the names -as they appear in the SYS.MACLIB(IGWSMF) macro -included within the text.