

## CICS Transactions Measurement with no Pain

Prepared by Luiz Eduardo Gazola  
 4bears - Optimize Software, Brazil  
 December 6 – 10, 2010  
 Orlando, Florida USA

*This paper presents a new approach for measuring CICS transactions reported on SMF type 110 Performance records throughout the use of z/OS Dynamic Exits facility to buffer the collected data in memory. The data is downloaded on real-time basis to SQL databases on open platforms via TCP/IP protocol. This innovative approach totally eliminates the need for DASD space and its associated I/O demands on the mainframe.*

### Introduction

CICS is by far the most commonly used IBM transaction monitor often supporting the execution of thousands of transactions per second and is present in virtually all large IT shops and data centers worldwide. In many online environments, CICS regions consume more CPU resources than any other component on the system however it lacks sufficient measurement facilities to address the following critical issues adequately:

- Average transaction response time;
- Resource usage; and
- Chargeback.

The following charts show examples of the typical behavior of seven different applications (transactions) during a 4-month period. Figure 1 presents the number of transactions by month, and figure 2 presents average transaction response times.

Figure 1 – Transactions per month

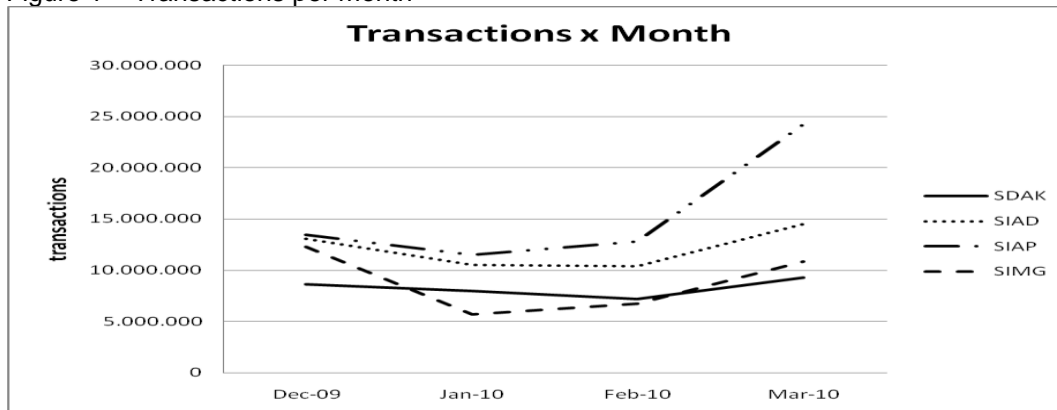
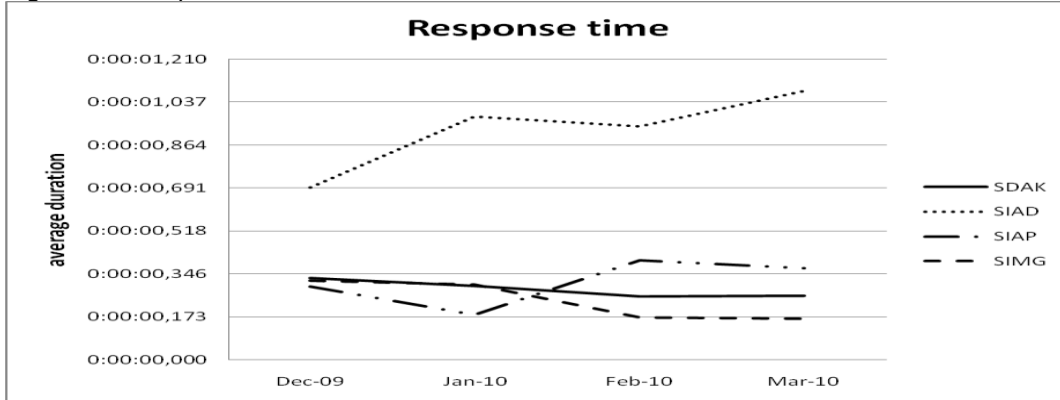


Figure 2 – Response times



Indeed, CICS measures the execution of each transaction execution in great detail; however, it does not consolidate or report the information so IT managers end up requiring additional software to measure and analyze such transactions.

Figure 3 shows a list of commonly used CICS monitoring and reporting software:

Figure 3 – CICS measuring products

Product	Vendor
<b>CICS Performance Analyzer (CICS PA)</b>	IBM
<b>Tivoli Decision Support for z/OS</b>	IBM
<b>MXG Software</b>	Merrill Consultants
<b>SAS IT Resource Manager</b>	SAS Institute
<b>CA JARS Resource Accounting</b>	Computer Associates
<b>CA MICS Resource Management</b>	Computer Associates

Unfortunately, all these products have the same problem: they require a lot of mainframe disk space and also use a considerable amount of I/O channel bandwidth for data movements in order to perform the measurements and consolidate the information.

The reason for the high volume of disk required is that CICS reports the details of each transaction execution such as date and time, issuer and CPU time usage, via the SMF type 110 records. Depending on the volume of transactions, this log can require an enormous amount of space just for the SMF raw records. In addition, consolidation of the information often demands a similar amount of DASD resources.

It is not surprising then that in the largest shops, these data requirements have become a storage management nightmare, requiring such a long time to process on daily basis that most IT managers prefer not to process this kind of information.

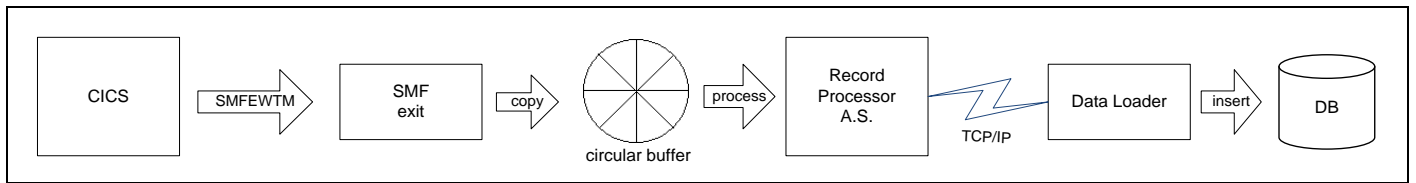
This paper proposes an alternative method to store and process this information out of the mainframe environment. In this approach, the records are intercepted before they are written to disk (while they are still in memory), then sent directly to a database on a less expensive platform.

### Proposed Model

The following sections describe the process used to intercept the SMF records as they are created, through the implementation of SMF exits. The SMF exit copies the contents of the records to another address space in order to process them in parallel, so there is no additional burden on the CICS regions due to I/O delays or any additional CPU consumption involved for the record processing. The exit also tells to SMF to discard the record.

The following diagram illustrates this process:

Figure 4 – Proposed model diagram



The record processor address space accepts the intercepted records and sends them to another process on the remote machine where they are inserted directly into the database.

### CICS Monitoring Facility (CMF)

CICS uses the SMF infrastructure to record all the transactions through the CICS Monitoring Facility (CMF) which create SMF type 110 Performance class records with the details of many transactions. The MCT (Monitoring Control Table) can be used to select the fields that will be recorded.

There are more than 200 fields which can be select for logging. Figure 5 shows some examples:

Figure 5 – Monitoring field descriptions

ID	Mnemonic	Description	ID	Mnemonic	Description
1	TRAN	Transaction code	71	PGMNAME	Program name
5	START	Transaction start time	89	USERID	User name
6	STOP	Transaction end time	167	SRVCLAS	WLM service class
2	TERM	Terminal identification	7	USRDISPT	Elapsed time on CICS

The performance class record is a multiple-transaction record; in other words, it holds as many transactions as possible provided the record does not exceed the 32 KB length limit. The CMF buffers the transaction counters in memory until an entire record is complete, then invokes the SMFEWMT macro to write the record.

Selecting a few fields on the MCT to reduce the volume of information saved on SMF can be a good practice. The default CICS configuration logs 2672 bytes of data per transaction, resulting in 12 transactions per SMF record. Selecting a smaller set of fields provides the ability to log more than 200 transactions on a single record quite easily.

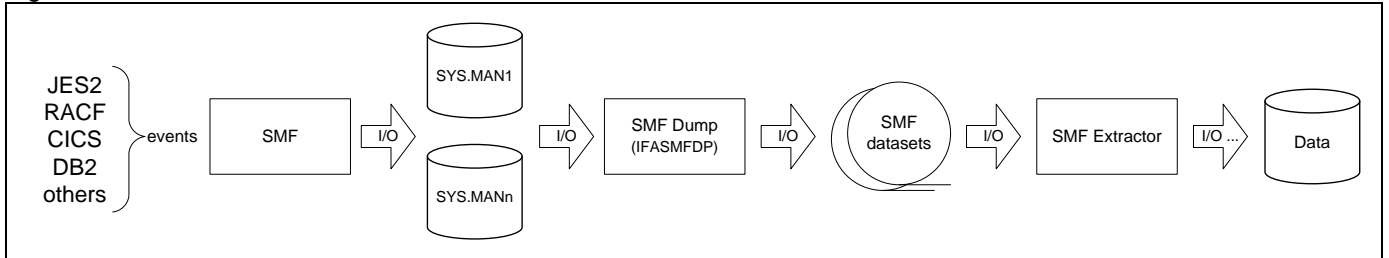
The monitoring can be activated dynamically through the following transaction: CEMT SET MONITOR ON PERF.

### SMF data collection

The System Management Facilities (SMF) is a z/OS component responsible for centralizing the records from various system components in a complex infrastructure where each component generates pre-formatted records. SMF then time-stamps the records and buffers them in memory. The records are written to VSAM datasets known as SYS.MANx files. When these files become full, they are copied (dumped) to sequential datasets which can be retained for many years.

Figure 6 illustrates a typical SMF processing flow:

Figure 6 – SMF data flow



- The logging components format a record according to its type and pass the record the SMF via the SMFEWTM macro;
- The SMF exits (IEFU8x) timestamp the records and write them to the SMF buffer in memory;
- The records are written to the VSAM datasets (SYS.MANx);
- When the SYS.MANx dataset is full, SMF switches to a new SYS1.MANx dataset and sends a console message to inform the operator that the file can be copied;
- The SMF Dump utility (program IFASMFDP) copies the VSAM dataset records to a sequential file (SMF dump files), in VBS (variable blocked spanned) format and empties the VSAM dataset for reuse;
- The SMF dump files are normally kept for very long periods of time.

The SMF records are the raw measurement and recording source for many software products and in-house applications and such records are usually read many times and retained for many years.

Some IT shops consolidate the SMF records on daily and monthly basis, merging all records of a certain period and LPARS on the same dataset to facilitate further data handling.

### Dynamic Exits and SMF

The Dynamic Exits Facility is another MVS component responsible for managing the connection of installation exit programs to their exit entry points dynamically and in a uniform way. The Dynamic Exits perform the following functions:

- Dynamic adding and removing of exit modules;
- Managing the chain of exit modules for each exit;
- Deactivating the exit module when it abends.

The exit module can be dynamically activated and deactivated through the SETPROG EXIT console command or programmatically using the CSVDYNEX macro.

SMF has three exits which are called every time the SMF writes a record: IEFU83, IEFU84 and IEFU85. These exits receive the entire SMF record and have the ability to alter its contents or even discard the record:

Figure 7 – SMF Records Exits

Exit name	Calling macro	Entry method
IEFU83	SMFWTM	Issue an SVC
IEFU84	SMFEWTM	Branch mode from primary address space
IEFU85	SMFEWTM	Branch mode from cross-memory

CMF issues the SMFEWTM macro in primary mode when a full record is complete in its buffer, so in order to intercept the performance records, the necessary code must be included in the IEFU84 exit. Figure 8 describes the available dynamic SMF exits for shops containing different SMF configurations.

Figure 8 – IEFU84 names

Exit name	Used for:
SYS.IEFU84	Default exit
SYSJES2.IEFU84	CICS started as batch with SUBSYS(JES2) on SMFPARMxx
SYSSTC.IEFU84	CICS started as started task and SUBSYS(STC) specified on SMFPRMxx

Keep in mind that the exit routine code has many restrictions and must be fast enough so as not to impact CICS due to processing delays. Normally, the best practice is to copy the contents of the record and process it in another address space.

Perhaps the easiest way to pass the record to another address space is via the buffers on Common Storage Area (CSA) which are common to all address spaces. Thus, a circular buffer in Extended CSA can be used to hold records temporarily until they are ready for processing.

The exit module should verify that the record is the performance class record, and if so, copy the contents to the buffer in common storage and suppresses record writing by placing a return code of X'04' in R15. Figure 9 illustrates the portions of the SMF type 110 records used to determine its subtype and class.

Figure 9 – Monitoring record

Field	Value
SMF Type	110
SMF Subtype	1 – Monitoring
Monitoring class	3 – Performance class

### Processing the Records

The performance records are multi-transaction records in which the field layout depends on the configuration defined on the Monitoring Control Table (MCT). They do not have any field delimiters and depend on the Dictionary record for their mapping.

The Dictionary record is also written as an SMF type 110 record. Every time the CICS starts or when the monitoring process is activated, a dictionary record is written. Dictionary records describe the fields present on the record, their type, size and offset. Therefore processing any performance record requires the corresponding dictionary record.

The dictionary record is identified by three key information fields: CICS name, CICS version and MCT modification version. Thus, changing the monitored fields in the MCT creates a new dictionary segment. Hopefully, every performance record will contain this key information. The following table describes the dictionary record key information fields:

Figure 10 – Dictionary record key fields

Field	Name	Description
SMF Type	SMFMNRTY	110
SMF Subtype	SMFMNSTY	1 – Monitoring
Monitoring Class	SMFMNCL	1 – Dictionary record
CICS name	SMFMNPRN	Generic CICS APPLID
CICS version	SMFMNRVN	CICS revision number
CICS modification	SMFMNMFL	Version of the MCT layout

The process can be simplified by using a common dictionary (MCT definition) for all CICS or alternatively, by saving all dictionary records in a dataset and loading them when processing the performance records.

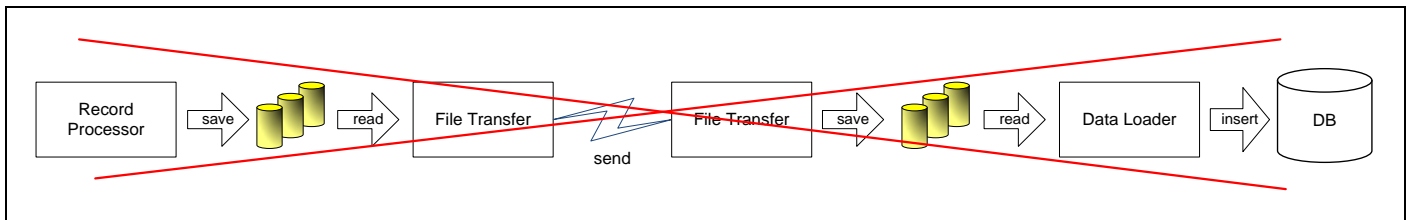
The records can either be processed on the mainframe before sending them to the remote system, or be sent in binary format and then processed remotely. If they are processed remotely, it will be necessary to perform some string conversions from EBCDIC to ASCII and also send the dictionary records. Processing them remotely can save some CPU cycles involved with converting the fields.

### Data transfer

The record processor runs in its own address space and can easily transfer the intercepted records to the remote system. One should be always cautious, however, to maintain enough space in the circular buffer into which the SMF exit writes the intercepted records so as avoid losing records.

The main idea is to send the records directly to the remote system without any disk access, such as what usually happens when data is written to disk. The file transfer process must then read the data from disk and send it to the remote file transfer program which writes the data on the remote disk where the data loader program reads the data again:

Figure 11 – File transfer model



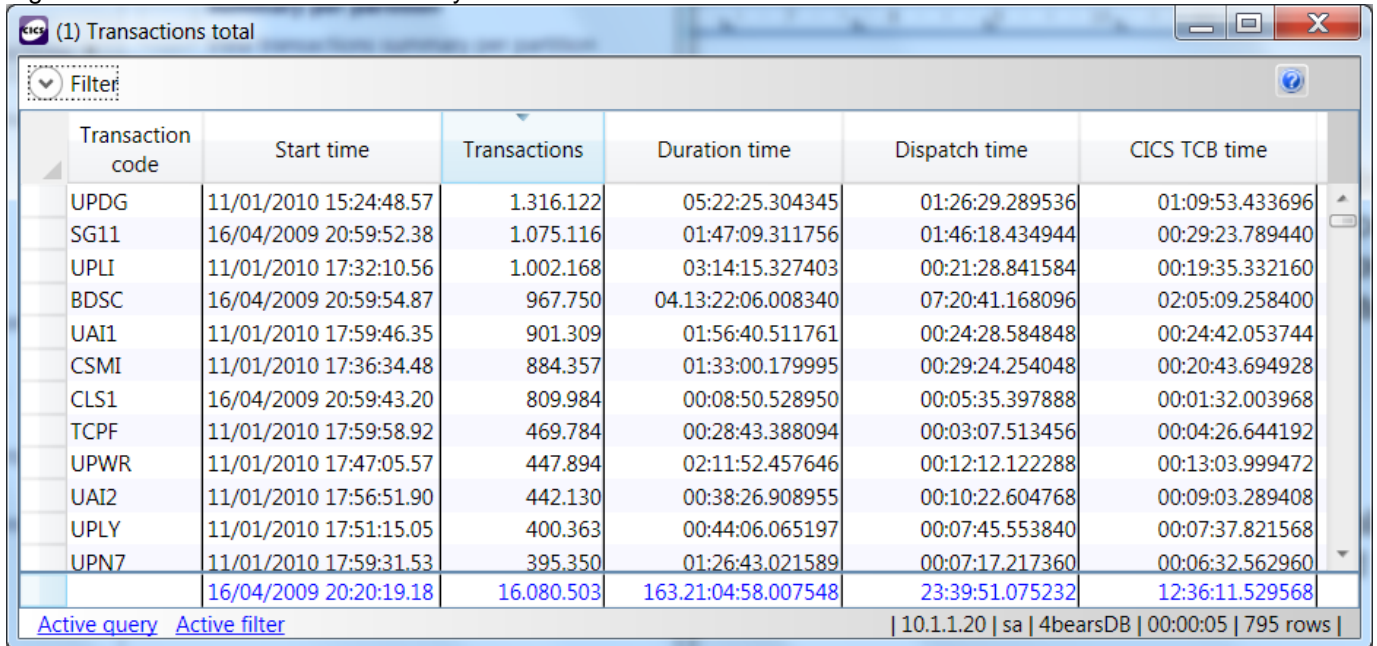
A much better solution would be a messaging facility such as IBM MQ-Series where the record processor can open a queue and send the data directly to the remote program, even in binary format. Also, protocols, connections and parallel producers and consumers on either side become non-issues.

### Remote system

On the remote system, the data loader program should receive the performance records and insert them into the database. Preparing the records to be inserted in bulk operations will improve performance. Some portability may be lost, but on the other hand, it is possible to load up to 10 thousand records per second.

Once the data is stored in the database it can easily be grouped into intervals and saved in historical tables. In our experiments the historical data occupies less than 5% of the original data. The following figure shows an example of transaction summary:

Figure 12 – CICS transaction summary



Transaction code	Start time	Transactions	Duration time	Dispatch time	CICS TCB time
UPDG	11/01/2010 15:24:48.57	1.316.122	05:22:25.304345	01:26:29.289536	01:09:53.433696
SG11	16/04/2009 20:59:52.38	1.075.116	01:47:09.311756	01:46:18.434944	00:29:23.789440
UPLI	11/01/2010 17:32:10.56	1.002.168	03:14:15.327403	00:21:28.841584	00:19:35.332160
BDSC	16/04/2009 20:59:54.87	967.750	04.13:22:06.008340	07:20:41.168096	02:05:09.258400
UAI1	11/01/2010 17:59:46.35	901.309	01:56:40.511761	00:24:28.584848	00:24:42.053744
CSMI	11/01/2010 17:36:34.48	884.357	01:33:00.179995	00:29:24.254048	00:20:43.694928
CLS1	16/04/2009 20:59:43.20	809.984	00:08:50.528950	00:05:35.397888	00:01:32.003968
TCPF	11/01/2010 17:59:58.92	469.784	00:28:43.388094	00:03:07.513456	00:04:26.644192
UPWR	11/01/2010 17:47:05.57	447.894	02:11:52.457646	00:12:12.122288	00:13:03.999472
UAI2	11/01/2010 17:56:51.90	442.130	00:38:26.908955	00:10:22.604768	00:09:03.289408
UPLY	11/01/2010 17:51:15.05	400.363	00:44:06.065197	00:07:45.553840	00:07:37.821568
UPN7	11/01/2010 17:59:31.53	395.350	01:26:43.021589	00:07:17.217360	00:06:32.562960
	16/04/2009 20:20:19.18	16.080.503	163.21:04:58.007548	23:39:51.075232	12:36:11.529568

Active query Active filter | 10.1.1.20 | sa | 4bearsDB | 00:00:05 | 795 rows |

### Conclusion

Nowadays, most typical mainframe workloads consist of online CICS transactions running around the clock; however, it is often a daunting and time-consuming task to figure out what is going inside the box. Unfortunately, most of the traditional measuring solutions add even more stress to the existing workload.

The new approach summarized in this paper has several advantages; substantially alleviating space and channel bandwidth demands on the DASD infrastructure; loading the transaction logs directly on the database, which is the best place to store such data; and reducing the CPU workload on the mainframe thus releasing resources that may be used to process even more CICS transactions.



## References

1. MVS System Management Facilities (SMF), IBM Corporation, SA22-7630-15, September 2007.
2. CICS TS for z/OS 4.1: Performance Guide, IBM Corporation, SC34-7033-00, 2009.
3. CICS TS for z/OS: CICS Customization Guide, IBM Corporation, SC34-6429-04, April 2007.
4. z/OS MVS Installation Exits, IBM Corporation, SA22-7593-12, September 2007.
5. MVS Programming: Authorized System Services Reference – Volume I (ALESERV – DYNALLOC), IBM Corporation, SC22-7609-08.